ENSEMBLES OF PROCESS-BASED MODELS OF DYNAMIC SYSTEMS

Nikola Simidjievski

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Prof. Dr. Sašo Džeroski, Jožef Stefan Institute, Ljubljana, Slovenia
Co-Supervisor: Prof. Dr. Ljupčo Todorovski, Faculty of Administration, University of Ljubljana, Ljubljana, Slovenia

Evaluation Board:
Prof. Dr. Juš Kocijan, Chair, Jožef Stefan Institute, Ljubljana, Slovenia
Acad. Prof. Dr. Ljupčo Kocarev, Member, Faculty of Computer Science and Engineering, “Ss. Cyril and Methodius” University, Skopje, Macedonia
Prof. Dr. Marko Robnik-Šikonja, Member, Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Nikola Simidjievski

ENSEMBLES OF PROCESS-BASED MODELS OF DYNAMIC SYSTEMS

Doctoral Dissertation

ANSAMBLI PROCESNIH MODELOV DINAMIČNIH SISTEMOV

Doktorska disertacija

Supervisor: Prof. Dr. Sašo Džeroski

Co-Supervisor: Prof. Dr. Ljupčo Todorovski

Ljubljana, Slovenia, May 2016

To my parents, my sister Marija and my aunt Nina

На моите родители, сестра ми Марија и тетка ми Нина

Acknowledgments

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, we had nothing before us, we had everything before us...

Charles Dickens
"A Tale of Two Cities"

Even after almost five years, I vividly remember the exact moment when I got the call from Prof. Dr. Sašo Džeroski, offering me the opportunity to work and do a doctorate in his research group. I can picture the silence in the room after the call, instantly broken by the sound of lighting a cigarette ... even my exact train of thought. It was one of those decisive moments filled with pride, curiosity and eagerness before embarking to a new exciting and life-changing journey accompanied with mild fear and sadness of leaving your home. In hindsight, standing before the end of this journey and ardently prepared for the next one, I never (not for a second) regretted my decision. Even though only my name is imprinted on the cover, nothing of this would have been possible without the support, both professional and personal, of a number of people.

My gratitude goes first to my supervisor Prof. Dr. Sašo Džeroski. Thank you Sašo – for making it all happen, for believing in my potential and providing constant support and many opportunities which earned me the necessary experience needed for scientific work. Your ideas, suggestions and unwavering enthusiasm for science kept me constantly engaged and focused on my research. My deep gratitude also extends to my co-supervisor Prof. Dr. Ljupčo Todorovski for his guidance, fruitful and inspiring discussions, encouragement and support which greatly influenced me and my work. Thank you Ljupčo – for your patience, for the criticism of my (un)successful ideas, thank you for accepting and sharing the failures and thank you for praising the success. You have been more than a mentor – you have been an excellent friend.

I would also like to thank the members of my PhD committee for carefully reading the manuscript of the thesis and providing useful feedback. Namely, Prof. Dr. Juš Kocijan, Acad. Prof. Dr. Ljupčo Kocarev and Prof. Dr. Marko Robnik-Šikonja offered constructive advice which greatly improved the quality of the work presented in the thesis. Most of the work presented in this thesis was financially supported by the Slovenian Research Agency through the Young Researchers Program. I would also like to acknowledge the financial support of the European Commission thorough the research projects SUMO – *Supermodeling by combining imperfect models* (ICT-2009-266722), MAESTRA – *Learning from Massive, Incompletely annotated and Structured Data* (ICT-2013-612944) and HBP – *The Human Brain Project* (ICT-2013-604102).

My deep gratitude also goes to my colleague, officemate and friend Dragi Kocev. Dragi – thank you for looking after me. Thank you for reminding me of my priorities and goals, and for selflessly sharing your experience and knowledge. I am grateful for all the criticism, comments and all of the debates and arguments (even those when we agreed to disagree). I am looking forwards to our next accomplishment. I am also grateful to Panče Panov, my colleague, coauthor and friend (and a short-time officemate), for all his advices, kindness and introductory lessons on *How to do a proper scientific work*. My gratitude also extends to my friend Aleksandra Rashkovska Koceva, for her support, advices and often (deeply appreciated) silent and indirect criticism which always somehow found the way to me. I would also like to especially thank my friend Ivica Slavkov. Ifko, you are the pinnacle of the category successful scientist with a successful social life. Thank you for reminding me that such a thing exists. Thank you for selflessly embracing a "rookie" like me, for all of the personal and professional advices and lessons (even when they were cynical) and, most importantly, for always being there for me wherever you are.

I would like to thank my colleagues from the *Department of Knowledge Technologies*, who provided a pleasant and friendly working environment as well as interesting scientific debates for various research problems. First, I would like to thank my officemates Jurica Levatić and Martin Breskvar. Djuro, thanks for the discussions, the shared frustrations and all of the support you gave me. Martin, thanks for the "slovenian" support and the "guys...I have a question" debates which undoubtedly kept me (and the other guys) engaged. You guys (together with Dragi) made every day feel like a Friday. Next, I would like to thank Jovan Tanevski, Darko Čerepnalkoski and Dragana Miljković who provided a pleasant atmosphere, both professional and social, through countless unforgettable moments of wisdom (and foolishness). I would especially thank Jovan, my friend, co-author and (often) brother-in-arms, for all the debates which resulted in ideas, comments and suggestion, but most importantly for sharing the load on every step of the way. Last but not least, I would like to thank Mili Bauer and Tina Anžič. You ladies are the backbone of the department, I am deeply thankful for your kindness, selfless help and advices during my studies.

I have been surrounded by many wonderful people in Ljubljana, to whom I also owe my deepest gratitude. First, to my friend Marija Drenkovska, for all of her unconditional moral support, kind and friendly advices and incredible exuberance which kept me motivated and my spirit high. I am also grateful to my friend Elena Ikonomovska, for her help and consult, but most importantly for showing me how to work for and achieve your goals. I would also like to thank Marija Crvenkovska, Vesna Andova, Vlado Kuzmanovski, Hristijan Gjoreski and Valentin Gjorgjioski for their support through a wonderful social atmosphere.

Living abroad is hard enough, so I would like to acknowledge several incredible friends from Skopje. Ivo Stojanoski, Nela Vasilevska, Andrijana Pavlovska, Bojan Milosheski, Aleksandar Josifovski, Katerina Chaloska, Andrej Siljanoski and Nikola Dimchev — from the day I told you I am leaving till the day I told you that I am almost over, you guys gave me constant long-distance support and warm embrace each time I visited home. For this, I am deeply thankful.

Finally, I am sincerely grateful to my family for their support, patience and understanding. Mom, Dad, Marija and Nina – I dedicate this work to you, since you have been the wind at my back at every step of the way. Your unconditional love and help gave me the strength and motivation to achieve this goal. Thank you — You are my foundation and none of this would have been achieved without you. Мамо, Тато, Марија и Нина: Неизмерно Ви благодарам за целата безрезервна поддршка, љубов и разбирање во ова мое патешествие.

It was the best of times.

Nikola
Ljubljana, 2016

Abstract

In this thesis, we propose a novel methodology for learning ensembles of process-based models, which addresses the task of automated modeling of dynamic systems. Mathematical modeling is at the heart of science and engineering and is often employed for representing and reasoning about the complex reality. Scientists and engineers construct complex mathematical models to study, control and/or predict the behavior of real-world systems under various conditions. Constructing mathematical models, however, is a nontrivial and demanding task that requires an extensive expert knowledge and empirical observations of the system at hand.

There are two main paradigms for modeling dynamic systems: theoretical (knowledge-driven) modeling and empirical (data-driven) modeling. In the former, a domain expert first derives a proper structure of the model, based on extensive knowledge about the system at hand. Measured data can then be used to estimate the (constant) parameters in the model, which, however, can also be set manually based on domain knowledge. The latter uses measured data in a trial and error process to search for such a combination of model structure/and parameters that best fits the observed behavior. Process-based modeling aims at integrating the theoretical and the empirical paradigms for modeling dynamic systems. More specifically, it aims at combining high-level process-based modeling knowledge and observed data from the system at hand. These are used for learning process-based models (PBMs) – where a PBM is a detailed, interpretable and modular representation of a dynamic system. The state-of-the-art process-based modeling approaches have been successfully applied to a variety of modeling tasks in a number of real-world domains. They have been predominantly used for describing the observed behavior of the modeled system. While focusing on the provision of detailed and accurate descriptions of the observed system, these approaches have only shown limited predictive power when applied to tasks of forecasting future system behavior.

To address this limitation of the process-based modeling paradigm, we focus on a well established approach for improving the predictive performance of models in machine learning — the paradigm of ensemble learning. Ensembles are a machine learning paradigm leading to accurate and robust models and are predominantly applied to predictive modeling tasks. Ensemble models comprise a finite set of predictive models, whose combined output is expected to yield an improved predictive performance as compared to the predictive performance of an individual process-based model.

The proposed methodology for learning ensembles of process-based models — the main contribution of this thesis — combines two lines of research. First, it extends the scope of existing approaches to learning process-based models of dynamic systems, predominantly employed in a descriptive modeling setting, towards predictive modeling. Second, it follows the basic principles of ensemble learning, as is common in machine learning, and translates them into a methodology for modeling dynamic systems. More specifically, we focus on a methodology for learning homogeneous ensembles of process-based models, which consist of models constructed by using the same machine learning algorithm on different samples of the training data and/or knowledge.

The proposed methodology includes algorithms for learning ensembles of process-based models by (1) bagging, i.e., learning from bootstrap samples of the data, (2) boosting, i.e., learning from error-weighted samples of the data, (3) learning from random library subsamples and (4) a combination of bagging and learning from random library subsamples. Furthermore, the methodology includes methods for simulating ensembles of process-based models that adapt the standard methods for combining predictions of ensemble constituents in machine learning. Overall, the proposed methodology provides a general and modular framework that allows for adapting other ensemble methods from machine learning to the particular context of learning process-based models.

We conduct an extensive experimental evaluation based on a novel empirical evaluation framework for process-based models of dynamic systems proposed in this thesis. This framework is employed for estimating the predictive performance of process-based models and ensembles thereof. This evaluation framework also identifies the optimal design choices within the proposed different ensemble methods. Furthermore, it allows for investigating whether the diversity among the individual ensemble constituents is related to the predictive performance of the ensemble. Finally, we analyze the computational efficiency of the proposed methods. These analyses relate to both the empirical evidence gathered from the performed experiments, as well as a theoretical study of the individual components in each of the proposed ensemble methods.

We apply the proposed methodology and evaluate its performance on a set of problems of automated predictive modeling of population dynamics in real-world aquatic ecosystems. The experimental results provide important evidence that ensembles of process-based models, in general, yield significantly better predictive performance compared to an individual process-based model. While the different proposed ensemble methods have similar predictive performance, the analysis of their computational efficiency reveals that ensembles learned from random library subsamples are the most efficient to learn. They are constructed using time comparable to the time needed to learn a single process-based model.

Povzetek

V doktorski disertaciji predstavljamo novo metodologijo za učenje ansamblov procesnih modelov, ki obravnava nalogo avtomatiziranega modeliranja dinamičnih sistemov. Matematično modeliranje je v samem jedru znanosti in inženirstva, kjer se spopada z nalogo opisovanja in razlage različnih resničnih sistemov. Znanstveniki in inženirji gradijo zapletene matematične modele z namenom, da preučijo, nadzorujejo in/ali napovedujejo odzive resničnih sistemov pod različnimi pogoji. Gradnja matematičnega modela je zahtevna naloga, ki mora upoštevati obsežno domensko znanje ter empirične ugotovitve o obravnavanem sistemu.

Obstajata dva poglavitna pristopa k modeliranju dinamičnih sistemov: teoretično modeliranje (na podlagi znanja) in empirično modeliranje (na podlagi podatkov). Pri prvem, domenski strokovnjak poda ustrezno sestavo modela, ki temelji na obsežnem znanju o obravnavanem sistemu. Za oceno (konstantnih) parametrov v modelu se kasneje lahko uporabijo meritve, parametre pa se lahko nastavi tudi ročno, na osnovi predznanja. Pri empiričnem pristopu, izmerjene podatke uporabimo za to, da z izkustveno metodo (ang. trial and error) poiščemo takšno kombinacijo sestave in parametrov modela, ki se kar se da dobro prilega opazovanemu sistemu. Cilj procesnega modeliranja je združitev teoretičnega in empiričnega pristopa, pri čemer kombiniramo visokonivojsko predznanje in meritve za učenje procesnega modela, ki je podrobna, razložljiva in modularna predstavitev obravnavanega dinamičnega sistema. Trenutno najboljše metode za procesno modeliranje so bile uspešno uporabljene v mnogih praktično relevantnih domenah za učenje opisnih modelov različnih sistemov. Kljub temu, da obstoječe metode preko modelov dajejo natančne opise obravnavanih sistemov, je napovedna moč dotičnih modelov šibka, kar negativno vpliva na zmožnost modela za napovedovanje obnašanja sistema.

Za premagovanje omenjene omejitve se pričujoče delo posluži uveljavljenega pristopa v strojnem učenju - pristopa ansambelskega učenja. Ansambelski pristop vodi k natančnim in odpornim modelom, ki so predvsem uporabljeni pri nalogah napovedovanja. Ansambelski modeli sestojijo iz končnega števila napovednih modelov. Pričakovati je, da bo združitev napovedi različnih modelov ustvarila boljše napovedi kot posamezni procesni modeli.

Predlagana metodologija za učenje ansamblov procesnih modelov, kar je tudi glavni prispevek te disertacije, združuje dva tira raziskav. Prvi nadgradi obstoječe pristope za učenje procesnih modelov dinamičnih sistemov, ki so največkrat uporabljeni v kontekstu opisnega modeliranja, v smer napovednega modeliranja. Drugi tir raziskav sledi osnovnim principom ansambelskega učenja in jih priredi v metodologijo za modeliranje dinamičnih sistemov. Osredotočimo se na metodologijo za učenje homogenih ansamblov procesnih modelov, ki sestojijo iz modelov, zgrajenih s pomočjo istega algoritma, iz različnih vzorcev podatkov v učni množici in/ali predznanju.

Metodologija vključuje algoritme za učenje ansamblov procesnih modelov z metodami (1) bagging, tj. učenje iz naključnih vzorcev podatkov, (2) boosting, tj. učenje iz z napako uteženih podatkov, (3) naključni knjižnjični vzorci in (4) kombinacija metode bagging z učenjem iz naključnih knjižničnih vzorcev. Nadalje, metodologija vključuje metode za simulacijo ansamblov procesnih modelov, ki uporabljajo običajne metode za združevanje napovedi posameznih pripadnikov ansambla. Predlagana metodologija predstavlja splošno in modularno ogrodje, ki omogoča uporabo ansambelskih metod iz strojnega učenja v kontekstu učenja procesnih modelov.

Opravili smo obsežno eksperimentalno primerjavo, ki je osnovana na novem empiričnem ogrodju za ovrednotenje procesnih modelov dinamičnih sistemov, ki je ravno tako predlagano v tej disertaciji. To ogrodje je uporabljeno za ocenjevanje napovedne moči procesnih modelov in ansamblov le-teh. Ogrodje za ovrednotenje ravno tako poda usmeritve za optimalno izbiro izmed predlaganih ansambelskih metod. Ogrodje dodatno omogoča tudi raziskovanje v smeri raznovrstnosti posameznih članov v ansamblu, kar pripomore k razumevanju o tem, kako posamezen član ansambla vpliva na napovedno moč celotnega ansambla. Za konec smo analizirali računsko učinkovitost predlaganih metod za učenje ansamblov. Omenjene analize se nanašajo tako na empirične meritve, pridobljene iz eksperimentov, kot tudi teoretične izračune kompleksnosti posameznih gradnikov v predlaganih ansambelskih metodah.

Predlagano metodologijo smo uporabili in ovrednotili na problemih avtomatskega napovednega modeliranja populacijske dinamike v vodnih ekosistemih. Eksperimentalni rezultati kažejo, da ansambli procesnih modelov v splošnem dajejo signifikantno večjo napovedno moč v primerjavi s posameznimi procesnimi modeli. Predlagane ansambelske metode imajo med sabo podobno napovedno moč, analiza računske učinkovitosti pa pokaže, da so ansambli naučeni iz naključnih knjižničnih vzorcev najbolj učinkoviti. Čas, ki je potreben za učenje takšnega ansambla, je primerljiv s časom učenja enega procesnega modela.

Contents

List of Figures xv

List of Tables xvii

List of Algorithms xix

I Introduction & Background 1

1 Introduction 3
- 1.1 Motivation . . . 5
- 1.2 Hypotheses and Goals . . . 8
- 1.3 Contributions . . . 14
- 1.4 Organization . . . 16

2 Background 19
- 2.1 Process-Based Modeling . . . 19
  - 2.1.1 Modeling dynamic systems – concepts . . . 19
  - 2.1.2 Automated modeling of dynamic systems . . . 21
  - 2.1.3 From equation rediscovery to process-based modeling . . . 23
  - 2.1.4 Process-based models . . . 25
- 2.2 Ensemble Learning . . . 32
  - 2.2.1 Ensemble combining schemes . . . 33
  - 2.2.2 Learning ensemble constituents . . . 35
  - 2.2.3 Interpretability of ensembles . . . 38
  - 2.2.4 Ensemble performance . . . 39
  - 2.2.5 Ensembles of models of dynamic systems . . . 40

II Methodology & Experimental Design 43

3 Ensembles of Process-Based Models 45
- 3.1 Simulating Ensembles of Process-Based Models . . . 46
- 3.2 Bagging of Process-Based Models . . . 47
- 3.3 Boosting of Process-Based Models . . . 48
- 3.4 Learning Process-Based Model Ensembles via Random Library Subsamples 50
- 3.5 Learning Process-Based Model Ensembles via Bagging of Random Library Subsamples . . . 51
- 3.6 Complexity of Constructing Ensembles of Process-Based Models . . . 52

4 Experimental Design 55
4.1 The Data . . . 55
4.2 Modeling Knowledge . . . 56
4.3 ProBMoT Settings . . . 58
4.4 Performance Evaluation Metrics . . . 58
4.4.1 Evaluating diversity . . . 59
4.4.2 Statistical comparison of performance . . . 59

III Results 61

5 Learning Ensembles via Sampling Data Instances 67

6 Learning Ensembles via Sampling the Library of Domain Knowledge 95

7 Learning Ensembles via Sampling Both Data Instances and the Library of Domain Knowledge 125
7.1 Design Choices . . . 125
7.2 The Performance of Ensembles of Process-Based Models Learned with the Bagging of Random Library Subsamples Method . . . 129
7.3 Predictive Performance and its Relation to Diversity . . . 130

IV Conclusions 135

8 Conclusions 137
8.1 Discussion . . . 138
8.1.1 On the performance of ensembles . . . 138
8.1.2 On the influence of diversity . . . 140
8.1.3 On modeling dynamic systems . . . 141
8.1.4 On ensemble interpretability . . . 142
8.2 Contributions to Science . . . 143
8.2.1 A novel methodology for learning ensembles of process-based models . . . 143
8.2.2 An evaluation framework for empirical analysis of the predictive performance of process-based modeling methods . . . 144
8.2.3 Improved long-term prediction as compared to state-of-the-art approaches to process-based modeling of dynamic systems . . . 144
8.2.4 Accurate long-term predictions of population dynamics in real-world aquatic ecosystems . . . 144
8.3 Further Work . . . 145

Appendix A Library of Domain Knowledge for Modeling Population Dynamics in Aquatic Ecosystems 147

References 155

Bibliography 165

Biography 167

List of Figures

Figure 1.1: General perspective of the research presented in this thesis. 5

Figure 1.2: Decomposition of the generalization error of (A) a single process-based model and (B) an ensemble of process-based models.[1] 6

Figure 2.1: Graphical representation of the relations (arrows and black boxes) between the entities (oval transparent boxes) in a simple lake ecosystem. . 26

Figure 2.2: Graphical representation of the algorithm for learning process-based models from knowledge and data 29

Figure 2.3: Simulation of phytoplankton concentration dynamics (solid line) as modeled with the process-based model from Table 2.2 and its comparison to observed phytoplankton concentration (dashed line). 31

Figure 2.4: Illustration of the Stack generalization (Stacking) method for creating heterogeneous ensembles. 34

Figure 2.5: Illustration of the Bootstrap aggregation (Bagging) method for creating homogeneous ensembles by sampling data instances. 36

Figure 2.6: Illustration of the boosting (AdaBoost) method for creating homogeneous ensembles by sampling data instances. 37

Figure 2.7: Illustration of the Random Subspaces method (RSM) for creating homogeneous ensembles by sampling data features. 37

Figure 3.1: A graphical representation of the time needed to learn a single process-based model and ensembles of process-based models with five constituents by bagging, boosting, library sampling, and bagging of library samples using the library presented in Table 2.1 from Section 2.1.4. . . 54

Figure 4.1: The library of modeling knowledge comprising template entities (thick horizontal lines) and processes (dashed lines connecting the entities) for modeling population dynamics in aquatic ecosystems. 57

Figure 7.1: Average ranks of ensembles with 10, 25 and 50 constituents combined by averaging and selected differently (based on their train (subscript T) or validation (subscript V) performance). The average ranks refer to the predictive (testing) model performance averaged over the 15 experimental data sets. 126

Figure 7.2: Average ranks of ensembles with 10, 25 and 50 constituents combined by averaging and selected differently (based on their train (subscript T) or validation (subscript V) performance). The average ranks refer to the descriptive (training) model performance averaged over the 15 experimental data sets. 127

Figure 7.3: Comparison of the average ranks for predictive performance with different ensemble sizes. Average ranks of ensembles that include 10, 25, and 50 base models in terms of predictive performance averaged over the 15 experimental data sets. 128
Figure 7.4: Average ranks of the four methods for combining the simulations of base models (average, weighted average, median and weighted median) in terms of predictive performance averaged over the 15 experimental data sets. 129
Figure 7.5: Average ranks of ensembles constructed by BRLS to the performance of: a single model and three types of ensembles constructed by bagging, boosting and RLS. The ranks of the models in terms of their predictive performance are averaged over the 15 data sets. 129
Figure 7.6: Scatter plots depicting the correlation between the diversity of the base-model predictions and the relative error improvement between a single model and an ensemble for fifteen data sets. The points colored in red denote the cases where ensembles perform worse than the average predictor (*ReRMSE* above 1). 133

Figure 8.1: An overview of the proposed ensemble methods (red) for learning ensembles of process-based models and a single process-based model (blue) in terms of predictive performance (*ReRMSE* obtained from the experimental evaluation) sorted with respect to increasing efficiency of the learning algorithms. The labels on the x-axis relate to relative computational time. 139

List of Tables

Table 2.1: Template entities and processes for modeling population dynamics in aquatic ecosystems. Here td(x) denotes the time derivative of x. 28
Table 2.2: A process-based model (top) of phytoplankton dynamics compiled to ODE (bottom), in the simple lake ecosystem from Figure 2.1, based on the template entities and processes from the library presented in Table 2.1. 31

Table 4.1: Overview of the data used for modeling population dynamics in the three lakes: Lake Bled, Lake Kasumigaura and Lake Zurich. 56

Table 7.1: Comparison of the predictive performance (RRMSE on the testing set) of the two methods for choosing ensemble constituents (V=validation, T=training). 126
Table 7.2: Comparison of the descriptive performance (RRMSE on the testing set) of the two methods for choosing ensemble constituents (V=validation, T=training). 127
Table 7.3: Comparison of the predictive performance of the BRLS method with 25 constituents combined by four different methods. 128
Table 7.4: Comparison of the predictive performance of a single model with that of ensembles of PBMs learned with BRLS, bagging, boosting and RLS. . . 130
Table 7.5: Diversity of the base models and the percentage of relative improvement of the ensemble error over the error of a single model learned for each of the fifteen experiments. In cases highlighted with an asterisk (*) the ensembles' performances are low (*ReRMSE* above 1). 132

List of Algorithms

Algorithm 2.1: Outline of the generic algorithm for learning process-based models from knowledge and data. 29

Algorithm 3.1: Simulating ensembles of process-based models. 46
Algorithm 3.2: Constructing ensemble of process-based models with bagging. . . . 47
Algorithm 3.3: Procedure for calculating the confidence of individual ensemble constituents. 48
Algorithm 3.4: Constructing ensemble of process-based models with boosting. . . . 49
Algorithm 3.5: Procedure for calculating the training sample of the boosting iteration. 49
Algorithm 3.6: Constructing ensemble of process-based models with random library subsamples 51
Algorithm 3.7: Constructing ensemble of process-based models with bagging of random library subsamples 52

Part I

Introduction & Background

Chapter 1

Introduction

Machine learning is an area in the realm of artificial intelligence (McCarthy, Minsky, Rochester, & Shannon, 1955), which studies algorithms with the ability to learn, i.e., algorithms that improve their performance through knowledge gathered from experience (Langley, 1996; Mitchell, 1997). Applications of machine learning algorithms can now be found in many different areas. These range from computational tasks in life sciences and earth sciences to social and behavioral sciences.

As in any algorithm there are two important ingredients necessary for its positive outcome: an input and an output. In the typical machine learning setting, i.e. inductive learning (Bratko, 2000), the input in a learning algorithm is (training) data which embodies the experience. The data consists of training examples (also referred to as instances or measurements) and their properties (also referred to as features or attributes). The properties can either describe the data or specify the desired output of the algorithm. Based on this, we can distinguish between descriptive properties and target properties, respectively.

On the other end, the output is defined by three components of the learning algorithm (Domingos, 2012): (1) task, (2) evaluation framework and (3) optimization criteria. The first component specifies the goal of the algorithm based on the provided data, where the second estimates its performance on the given task. Note, however, machine learning is an ill-posed problem. This means that there is no unique solution (model) as an output of a learning algorithm. Most often, the output consists of many solutions, and the purpose of the last component is selecting the proper one. Simply put, the output is a model which estimates the performance of a learning algorithm on a specific task.

In the general layout of machine learning tasks we can divide them based on their purpose and based on available data provided, i.e., output and input. Based on their purpose (output) we can clearly distinct between descriptive and predictive tasks. The former group includes tasks that aim at learning a model for identifying patterns and obtaining general descriptions of the training data. In contrast, the latter group is comprised of tasks the goal of which, is to obtain a model that can predict properties of new instances based on an experience obtained in the training process.

Machine learning tasks can also be roughly grouped based on the type of available data (input) provided. Here we can distinguish between supervised and unsupervised learning tasks. If the data provided at input consists of training examples and both their descriptive and target properties, supervised learning tasks aim at obtaining a model which is a general rule associating the input to the desired output. On the other hand, unsupervised learning involves a class of tasks which tackle the problem of finding general rules and descriptions of the provided data where the target properties of the training examples are not provided.

Literature on machine learning (Langley, 1996; Mitchell, 1997; Bishop, 2007) often aligns the classes of unsupervised and supervised tasks with the classes of descriptive and predictive tasks, respectively. However, these two classifications are orthogonal (Flach, 2012). There are: (1) descriptive unsupervised learning tasks eg. (descriptive) clustering (Kaufman & Rousseeuw, 1990), principal component analysis (Jolliffe, 2006) and association rule learning (Agrawal, Mannila, Srikant, Toivonenu, & Verkamo, 1996), (2) descriptive supervised learning tasks eg. subgroup discovery (Lavrač, Kavšek, Flach, & Todorovski, 2004) and equation discovery (Langley, Simon, Bradshaw, & Zytkow, 1987; Bridewell, Langley, Todorovski, & Džeroski, 2008), (3) predictive unsupervised learning tasks eg. predictive clustering (Blockeel, Raedt, & Ramon, 1998; Kocev, Vens, Struyf, & Džeroski, 2013) and (4) predictive supervised learning tasks eg. classification and regression (Breiman, Friedman, Stone, & Olshen, 1984).

This thesis addresses the task of predictive supervised learning. In machine learning there is an ample amount of methods successfully employed for predicting purposes in a number of real life domains (Breiman et al., 1984; Witten & Frank, 2005). They have been applied for predicting discrete output values (classification), continuous output values (regression), even structured outputs as in gene networks, image classification, text categorization etc. (Bakır et al., 2007). A small subset of such methods includes decision and regression trees (Breiman et al., 1984), neural networks (Bishop, 1995), support vector machines (Cortes & Vapnik, 1995) etc. Naturally, the question here is choosing the most suitable method. The straightforward approach is choosing a method which produces a model with a good generalization power. Generalization power or predictive performance is the ability of the learned model to accurately predict the property of a new (unseen) instance beyond the data used in the training process. However, as simple as this may seem, many machine learning methods are prone to overfitting, struggling with obtaining a suitable predictive model. Overfitting occurs when a method learns a model with a good performance on the provided training data, subsequently limiting its generalization power.

Machine learning approaches address the problem of overfitting typically as a part of the optimization criteria component of the learning algorithm. Such approaches include: cross-validation, regularization and various statistical tests (Kohavi, 1995). Among them, there is also a long tradition of developing methods for learning multiple models and combining their outputs instead of just learning a single model. These methods are referred to as ensemble methods or ensembles. Ensembles are a well-accepted formula for improving the predictive performance of models in machine learning (Dietterich, 2000b). An ensemble is a set of models that is expected to lead to predictive performance gain over an individual model, by combining the predictions of the ensemble constituents (Dietterich, 2000a; Breiman, 2001; Freund & Schapire, 1999; Maclin & Opitz, 1999; Bauer & Kohavi, 1999). It is important to stress out that ensembles tend to alleviate the overfitting not eliminate it. There are many theoretical and empirical studies that demonstrate their predictive ability and superiority over single models on a plethora of predictive tasks (Kuncheva, 2014; Seni & Elder, 2009; Banfield, Hall, Bowyer, & Kegelmeyer, 2007; Levatić, Ceci, Kocev, & Džeroski, 2015).

The work presented in this thesis combines two different lines of research. First, it follows the basic principles of ensemble learning, and translates them into a methodology for modeling dynamic systems. Second, it extends the state-of-the-art paradigm of equation discovery, which is predominately employed in descriptive setting towards predictive modeling. More specifically, we illustrate the utility of learning ensembles of process-based models for long-term prediction of the behavior of dynamic systems. The general perspective of the research presented in this thesis is depicted in Figure 1.1.

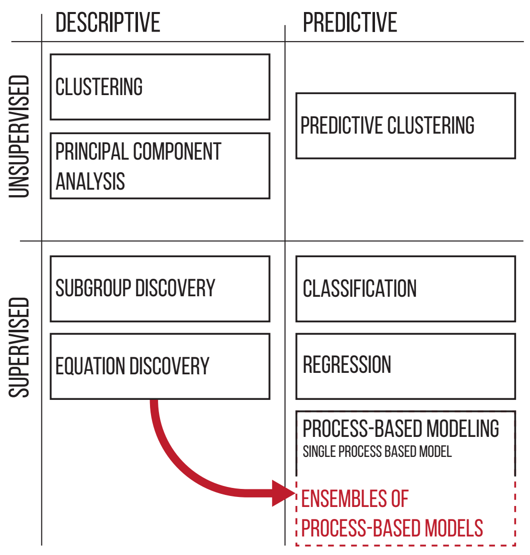


Figure 1.1: General perspective of the research presented in this thesis.

1.1 Motivation

Mathematical models of dynamic systems are constructed and employed to provide an understanding of the nature laws that govern the behavior of the system at hand. More specifically, such models are being utilized to recreate or simulate the behavior of dynamic systems under various conditions (Ljung, 1999; Strogatz, 1994). There are two main pillars of every approach to modeling a real dynamic phenomena: (1) structure identification and (2) parameter estimation. The former, address the task of establishing a suitable structure (commonly an equation) of the model in terms of defining the components that are involved in the system and how they interact. The latter, deals with approximation of the constant parameters and the initial values of the variables in the model for the given structure.

In general, there are two main paradigms for modeling dynamic systems. The major paradigm used when modeling dynamic systems is the approach of theoretical (knowledge-driven) modeling. Here, a domain expert first derives a proper structure of the model based on an extensive knowledge about the system at hand. Measured data are then used to estimate the (constant) parameters in the model. The alternative paradigm, i.e. empirical (data-driven) modeling, uses measured data in a trial-error process to search for such a combination of model structure/parameters that best fit the measurements.

Equation discovery (Langley et al., 1987), a sub-field of machine learning, joins the previous two paradigms of modeling, by studying methods for learning both model structure and parameter values from observations (Džeroski & Todorovski, 1993; Todorovski & Džeroski, 2007). These methods simultaneously tackle the tasks of structure identification and parameter estimation. The most recent approach, referred to as process-based modeling, combines heuristic search methods with parameter estimation techniques. While the search methods explore the space of candidate model structures, the parameter estimation techniques find optimal values of the parameters of each candidate structure and evaluate its fit against the measured data.

The resulting process-based models provide a high-level explanatory structure and a low-level mathematical formulation which allows for simulation of the model under various conditions (Džeroski & Todorovski, 2002; Bridewell et al., 2008; Todorovski, Bridewell, Shiran, & Langley, 2005). The former is a direct result from employing modeling knowledge, formulated in a process-based domain library, which describes the basic types of components that can occur in the modeled system and additionally provides alternative modeling recipes for each of them. The latter relates to incorporating ordinary differential equations (ODEs). ODEs are a widely accepted formalism for modeling dynamic systems, which define the rate of (typically temporal) change of the system variables in a dynamic system. All in all, process-based models bridge and build upon purely empirical and purely theoretical approaches to modeling, and strive at explaining how and why the dynamic system behaves under various conditions.

The utility of the process-based modeling approach has been demonstrated in a variety of modeling tasks addressing population dynamics in lake (Atanasova, Todorovski, Džeroski, Remec, et al., 2006; Atanasova, Recknagel, Todorovski, Džeroski, & Kompare, 2006; Čerepnalkoski, Taškova, Todorovski, Atanasova, & Džeroski, 2012; Taškova, Šilc, Atanasova, & Džeroski, 2012) and marine (Bridewell, Asadi, Langley, & Todorovski, 2005; Bridewell et al., 2008) ecosystems, watershed modeling (Škerjanec, Atanasova, Čerepnalkoski, Džeroski, & Kompare, 2014), modeling endocytosis (Tanevski, Todorovski, Kalaidzidis, & Džeroski, 2015; Taškova, Korošec, Šilc, Todorovski, & Džeroski, 2011), gene regulatory networks, epidemiological dynamics (Tanevski, Todorovski, & Džeroski, 2016) and dynamics of biological networks (Džeroski & Todorovski, 2010). However, these applications focus primarily on establishing explanatory models of the system at hand. Typically, the obtained models are simulated and analyzed on the same data used for learning them, thus their ability to predict the systems' behavior is limited, i.e., their generalization error has not even been considered nor evaluated. Moreover, the results of several preliminary experiments indicate that process-based models in the respective studies have a tendency to overfit: Their ability to accurately describe the observed behavior of the modeled system limits their generalization power, subsequently impairing their potential to accurately predict future system behavior.

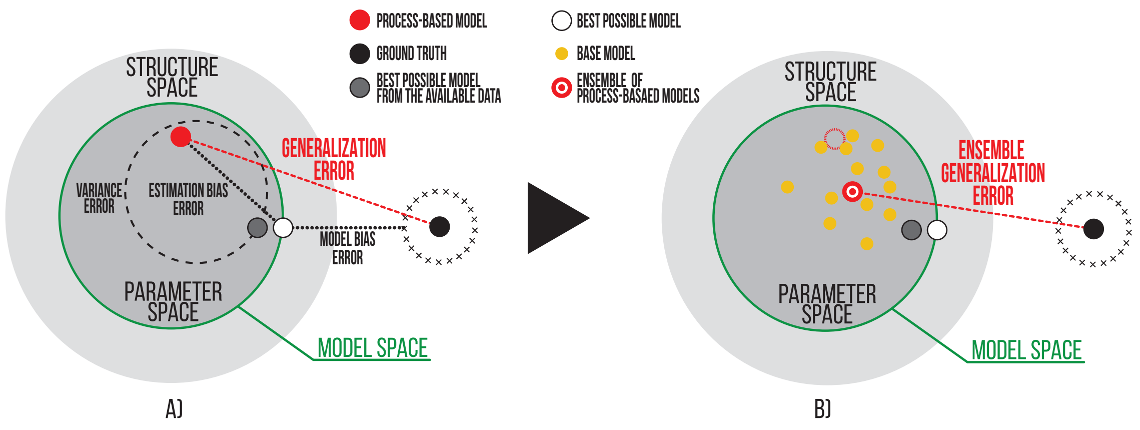


Figure 1.2: Decomposition of the generalization error of (A) a single process-based model and (B) an ensemble of process-based models.[1]

Figure 1.2A illustrates this problem by decomposing the generalization error (red dashed line) between a process-based model (red point) and the ground truth (black point). In the space of possible structures and parameters, suppose we learn a model that accurately describes the observed behavior of a dynamic system. This model, however, may

[1]The concept of the illustration is borrowed from Kuncheva (2014) and appropriately adapted for the purpose of the thesis.

still be far from the optimal model (gray point) in the same model space (green stroke), determined by the training data and domain knowledge provided as input. This is referred to as a *variance error*. In general, such error usually can occur from the inability of the learning algorithm (limited by the training data) to converge towards a global optimal solution in terms of best structure-parameters combination. For the task of learning process-based models, this is even more amplified, given that such models rely on a large number of parameters and generally are highly unstable (their output can significantly vary with small variations in the input) (Breiman, 1996a).

However, the potentially optimal model that can be learned may still be far from the best possible solution (white point) of the modeling task at hand, which in practice is hard to achieve. This discrepancy is referred to as an *estimation bias error*, and is an artifact from the amount of data used in the process of learning a process-based model. Looking at the bigger picture, there is a discrepancy between the best possible solution and the truth. This is referred to as *model bias error*, and is determined by the amount of modeling knowledge (modeling components) used in the process creating the model space. Finally, the last part of the generalization error (depicted with crosses) is the irreducible error. This error, however, is a result of a fundamental limitation of the whole concept of modeling a real system, which occurs even if (ideally) infinite amount of data is provided: In other words, there is no perfect model.

The research presented in this thesis addresses the afore-mentioned limitation of the process-based modeling paradigm in terms of generalization error by extending its scope towards learning ensembles of process-based models. More specifically, we propose a novel methodology for learning ensembles of process-based models employed for predictive modeling of dynamic systems. Figure 1.2B illustrates the aim of this thesis. As we stated earlier, the intuition behind constructing an ensemble is to alleviate the generalization error by combing the predictions of individual base models (orange points), subsequently yielding a better predictive performance. These base models, however, should at least be moderately accurate (at least perform better than random guessing) and diverse (should err differently and independently).

Ensembles of models of dynamic systems, have been previously considered in two different fashions. First, the approach presented by Bridewell et al. (2005) considers ensembles of process-based models, where the structures of the ensemble constituents are integrated into a single meta-level model. The ensemble constituents are learned from different random training data samples. The single meta-level model is built in such a way that it includes the most frequent structure fragments (processes) in the base-level models. The results show that the resulting meta-level model still provides a process-based explanation of the observed system structure, while being more robust in terms of over-fitting. Note, however, that the authors estimate the out-of-sample error of the models by taking random sub-samples of the observed time-series data and removing them from the training data. Thus, the ability of the meta-level model to generalize outside the time span of the training data has not been considered nor evaluated.

Second, Aleksovski, Kocijan, and Džeroski (2015) address the tasks of predictive modeling of discrete non-linear dynamic systems. Here, using external dynamic approach, the modeling problem is first transformed into a non-linear regression approximation problem. The system then is modeled by learning fuzzy linear model trees and ensembles of fuzzy linear model trees. The results show that ensembles of fuzzy linear model trees can be employed for the task of modeling dynamic systems and , more importantly, improve the predictive performance over the single fuzzy linear trees. Note that, this study focuses on short-term (one-step-ahead) prediction of discrete-time dynamic systems, where the value of a time series in the next time-point is predicted, and then the predicted value is fed back to the model for predicting the next one.

Finally, in a broader sense, ensemble predictions of dynamic systems are achieved by combining diverse predictions from an individual model. These predictions are obtained either by perturbing the initial states of the model or varying (some of) the constant parameters in the model. A related recent development is the paradigm of supermodeling (Van den Berge, Selten, Wiegerinck, & Duane, 2011; Mirchev, Duane, Tang, & Kocarev, 2012). A supermodel is comprised from a set of interconnected (coupled) models, which are integrated simultaneously and exchange information among themselves on a time-step basis. The supermodels, albeit fairly complex and consisting of typically small number of constituents (which are provided by a domain expert rather than being learned), have been successfully employed in the context of climate modeling for addressing the task of long-term climate projections.

The proposed methodology in this thesis borrows some ideas from well-established ensemble methods in machine learning, however it distinctively differs in several important aspects. First, it differs in the composition of the obtained ensembles. Such ensembles are comprised from process-based models learned either from different samples of the measured data, samples of the library of domain knowledge or both. We conjecture that this will improve the generalization ability of ensembles of process-based models, subsequently leading to improved predictive performance. Second, it differs in the output obtained from the ensembles of process-based models: While traditional ensembles, applied in the context of time-series, are typically used for short-term prediction of the state at the next time point, based on the observed values of the current and previous states — the ensembles of process-based models can provide long-term predictions over many following time points, relaying only on the initial values of the state variables.

All things considered, the problem that we propose to investigate in this thesis is: *Can we employ the ensemble methodology from traditional machine learning for modeling dynamic systems, and more importantly, does it yield an improved predictive performance over the state-of-the-art process-based modeling methods?*

1.2 Hypotheses and Goals

This dissertation will tackle the task of predictive modeling of dynamic systems by learning ensembles of process-based models. Our hypotheses are formulated as follows:

Hypothesis 1. *Traditional machine learning ensemble methods can be adapted to the task of learning process-based models.*

Hypothesis 2. *Ensembles of process-based models yield improved predictive performance compared to a single process-based model.*

Hypothesis 3. *The diversity between ensemble constituents is highly correlated with the predictive performance of an ensemble.*

In order to properly prove the afore-stated hypotheses, we set a number of more specific goals which address three key aspects of the task of learning ensembles of process-based models: design, implementation and evaluation. The goals are defined as follows:

Goal 1. Design – *Design a methodology for simulating ensembles of process-based models.*

Goal 2. Design – *Design a process-based modeling methodology for generating a diverse set of ensemble constituents.*

Goal 3. Design – *Develop an evaluation framework for testing the utility of the proposed approaches to learning ensembles of process-based models.*

Goal 4. Implementation – *Implement a methodology for constructing and simulating ensembles of process-based models and an evaluation framework for testing their predictive performance.*

Goal 5. Evaluation – *Perform extensive empirical analyses to obtain optimal design choices for each of the proposed methods for constructing ensembles of process-based models.*

Goal 6. Evaluation – *Conduct a thorough and fair empirical evaluation of the predictive performance of the different proposed ensemble methods.*

Goal 7. Evaluation – *Measure the influence of the intra-ensemble diversity on the predictive performance of the ensemble.*

The specific details of the goals outlined above are summarized in the following subsections.

Design a methodology for simulating ensembles of process-based models

When designing a method for learning an ensemble of process-based models, or ensembles in general, there are two important questions that should be addressed: (1) how the output of the ensemble will be generated and (2) how the individual ensemble constituents will be constructed. The first milestone when constructing ensembles is choosing the most appropriate technique for combining predictive models. There are two general approaches for generating the output of the ensemble: model combination (or fusion) and model selection (Džeroski, Panov, & Ženko, 2009). In the latter approach, each base model is first evaluated and the prediction of the best performing one is used as the overall ensemble prediction. The ensembles combined with this approach are interpretable and usually computationally efficient for learning, however their predictive performance is limited by the performance of the selected model.

In contrast, the former and more sophisticated approach uses different combining schemes for aggregating the predictions of all ensemble constituents into an ensemble prediction. Based on how these combining schemes are employed for generating the ensemble output, we can further distinguish between dynamic and fixed combining schemes. The former includes learning the scheme of how the ensemble constituents will be aggregated (Wolpert, 1992). The latter, fixed (a posteriori) combining scheme, relates to first learning the base models, and then employing some combination rule for aggregating the individual predictions. For the task of predicting numeric values, the most widely (and simplest) fixed scheme used is averaging: The mean value of every base-model prediction represents the overall ensemble output. Moreover, a weighted variant can be used, where the assigned weights correspond to the performance (MSE, RMSE, ReRMSE, model confidence) of each of the base models (Drucker, 1997).

In this thesis, for simulating an ensemble of process-models, we will employ standard fixed combination approaches used in machine learning for constructing ensembles. More specifically, when designing the simulation methodology we will take note of the similarity between the task of process-based modeling and time-series regression and focus on approaches such as: averaging, median and their weighted variants. However, in contrast to the task of obtaining an output from a regression model, where the resulting prediction is a single point for a given input, the task of predicting with process-based models is far more challenging. The simulation of a process-based model takes a minimal input consisting of the initial values of the state variables and the complete trajectories of the forcing variables. As output, it produces complete (long-term) trajectories of the state variables. In a predictive scenario, this can often lead to divergent trajectories and disastrous predictive misperformance. To alleviate this, we will also aim at investigating ensemble pruning techniques, which will potentially improve the robustness of the ensemble, subsequently leading to more stable and accurate predictions.

Design a process-based modeling methodology for generating a diverse set of ensemble constituents

The second milestone in constructing ensembles is designing the algorithm(s) for learning the ensemble constituents. Based on how the constituents are learned, the ensembles can be heterogeneous or homogeneous. In heterogeneous ensembles, the candidate base models are learned using different learning algorithms (Wolpert, 1992). In contrast, homogeneous ensembles consider base models that are learned with the same learning algorithm, but from different training samples generated by manipulating the training data. These manipulations of the training data include (1) sampling of data instances (as in bagging (Breiman, 1996a) and boosting (Schapire & Freund, 2012)), (2) sampling of data features/attributes (as in random subspaces (Ho, 1998)) or (3) both (as in random forests (Breiman, 2001) and bagging of random subspaces (Panov & Džeroski, 2007)).

Given that in this thesis we rely on one algorithm for learning process-based models, the aim is investigating and designing methods for learning homogeneous ensembles of process-based models. Bagging (bootstrap aggregation) refers to the approach, developed by Breiman (1996a), for constructing ensembles via bootstrap sampling with aggregation. This is one of the first and simplest ensemble learning methods, where data instances are uniformly sampled with replacements to generate random samples (bootstrap replicates) of the training data, consequently used for learning the ensemble constituents from these replicates.

Boosting refers to a general approach for obtaining an accurate prediction by combining several less accurate ones learned on different distributions of the training data. The AdaBoost algorithm, proposed by Freund and Schapire (1997), is one of the most employed boosting approaches. AdaBoost works iteratively; it uses different distributions of the training data for learning the base models at each iteration. Depending on the outcome of the past iteration this method decreases (for correct classification)/increases (for incorrect classification) the weights of every instance, thus changing the distribution for the subsequent iteration of training the model. In this way, the individual weak predictors focus on different instances, and their combination is more accurate. In a similar fashion, the implementation of Drucker (1997) successfully tackles the problem of combining regressors using AdaBoost. Several studies show that homogeneous ensembles, such as bagging and boosting, perform well for regression problems (Bauer & Kohavi, 1999; Breiman et al., 1984; Freund & Schapire, 1999; Dietterich, 2000a). However, these methods, that sample data instances, can often be ineffective when the training data is relatively homogeneous. Moreover, when the dimensionality of the data (the feature space) is very high, learning such ensembles can be very ineffective and computationally complex.

The random subspace method (RSM) is a homogeneous ensemble method developed by Ho (1998), which constructs different variants of the training data by sampling the feature space. Each ensemble constituent is learned on all data instances and a subspace of the original feature space. The RSM has been reported to perform well for problems where the data dimensionality is very high or when there is a certain redundancy in the feature space (Ho, 2000). Finally, a study performed by Panov and Džeroski (2007) indicates that combining two types of homogeneous ensembles, i.e. bagging and RSM, can lead to ensembles with even better predictive performance compared to each of the methods separately, while still being very efficient to construct.

All things considered, in this thesis, we propose four methods for learning ensembles of process-based models. The first two methods will focus on learning ensembles by sampling the data instances. More specifically, we intend to develop methods which will follow the key principles of *bagging* and *boosting* (more specifically Adaboost) in the context of time-series regression. The third method will consider learning ensembles of process-based models by sampling data features. In this context, we can think of the feature space as being defined by the components instantiated from the process templates. This space of components is determined by the number of process alternatives defined in the library of domain knowledge. Therefore, the task of generating random samples of the feature space from the traditional RSM, has as an analog to the task of generating random subsamples of the library of domain knowledge. We refer to this method as the *random library subsamples* (RLS) method. Finally, given the nature of the ensembles of process-based models learned with random library subsamples and bagging (i.e., sampling the feature space and sampling the data instances), we plan to develop a method which combines them, thus constructs ensembles from different samples of the data and the library. We refer to this method as *bagging of random library subsamples* (BRLS).

Develop an evaluation framework for testing the utility of the proposed approaches to learning ensembles of process-based models

To test the validity of our ensemble methodology, we intend to perform extensive empirical analyses of the implemented methods on the task of modeling and predicting population dynamics in aquatic ecosystems. The experiment evaluation will involve a series of tasks of modeling dynamics in real-world ecosystems. It includes three real-world lakes: Lake Bled in Slovenia, Lake Kasumigaura in Japan and Lake Zurich in Switzerland. For all the real-world ecosystems we intend to use the same structure of population dynamics model. It includes a single equation (ODE) for a system variable representing the phytoplankton biomass, and several exogenous variables that represent other populations in the ecosystems (zooplankton), inorganic nutrients and environmental influences.

The aim of such an evaluation framework is two-fold. First, it will allow us to identify the optimal design properties of the different ensemble methods we propose in this thesis. Second, it will provide us a thorough and fair evaluation of the predictive performance of the proposed methods. The second aspect of such an evaluation framework directly relates to addressing the first two hypotheses proposed in this thesis. More specifically, besides comparing every proposed ensemble method to a single model in terms of long-term predictive performance, we also seek for the most appropriate ensemble method applied in the context of process-based modeling. Finally, to tackle the last proposed hypothesis, we will investigate whether the diversity among the individual ensemble constituents is related to the predictive performance of the ensemble.

Implement a methodology for constructing and simulating ensembles of process-based models and an evaluation framework for testing their predictive performance

The state-of-the-art implementation of the process-based modeling paradigm is the Process-Based Modeling Tool (ProBMoT) (Čerepnalkoski, 2013), a software platform for construction, parameter estimation, and simulation of process-based models. ProBMoT follows the process-based modeling principles in terms of addressing the two main tasks of modeling dynamic systems[2]: (1) structure identification and (2) parameter estimation. It employs domain-specific modeling knowledge represented in a library and introduces incomplete conceptual models as modeling assumptions, in order to constrain the space of model alternatives in terms of model structures and parameter constraints. Using the library of domain knowledge and modeling assumptions, ProBMoT first enumerates all plausible model structures, subsequently estimating the parameters of each of them using measured data of the modeled system. To this end, ProBMoT implements the state-of-the-art global optimization methods for parameter estimation and allows for the use of a variety of different objective functions. The output consists of a list of complete process-based models, ranked according to their performance measured on the training data.

In this thesis, we will extend the ProBMoT towards constructing, simulating and evaluating ensembles of process-based models. These extensions directly relate to the design goals outlined previously. First, in terms of a methodology for simulating ensembles of process-based models, the ProBMoT extensions include implementation of different techniques for combining simulations of multiple process-based models. Given that a simulation of process-based model consists of numeric values, the techniques we will focus on include: average, median, weighted average and weighted median. The numeric-valued predictions of the constituent models will be combined per time-point, for each time-point separately. To account for the divergent trajectories that may occur when simulating the individual base models, we intend to implement a knowledge-driven pruning technique that takes into account the constraints provided in the library of domain-knowledge.

Second, the implementation of the proposed four different ensemble methods includes extending ProBMoT in two directions, i.e. extending the methods that tackle structure identification and parameter estimation. More specifically, the method for *bagging* process-based models includes implementing new objective function (Weighted Root Mean Square Error) which will be able to handle learning different models from different (randomly generated) bootstrap replicates of the training data. In the case of *boosting* of process-based models, we will use the same objective function, however in a slightly different fashion. Instead of learning the ensemble constituents from random samples with replacements of the training data, here each model will be learned on a sample of the training data generated based on the performance of the preceding model in the sequence. This involves implementing a confidence function, which will be applied to every ensemble constituent in the ensemble. Moreover, we will use this same function when combining the base models using the weighted combining schemes.

Implementation of the method for learning ensembles by sampling the library of domain-knowledge, i.e., the *random library subsamples* (RLS) method, involves extending the ProBMoT's methods that address the structure identification task. Earlier we mentioned that the task of generating random samples of the feature space has as an analog to the task of sampling the space of model components determined by the number of process alternatives. Therefore we intend to implement a sampling algorithm which will generate libraries with different sizes (i.e, libraries which can lead to a different number of learned models) that will be used in the process of learning the individual ensemble constituents. For the last method, i.e., *bagging of random library subsamples* (BRLS), we intend to combine all the previous implementations, and construct ensembles from different samples of the training data and the library of domain knowledge.

Finally, considering the empirical evaluation framework of the proposed ensemble methodology, we plan two important extensions to ProBMoT. In the current implementation,

[2]A more detailed overview of the process-based modeling paradigm is given in Section 2.1 of Chapter 2

ProBMoT supports only one (the same) dataset for learning and evaluating the learned process-based models. The first extension includes adding a multi-dataset support to ProBMoT, which will give us the ability to train,validate and test the ensembles on different datasets, thus thoroughly evaluate their performance. The second extension involves employing the performance measures. To this end, we plan to observe and compare the predictive performance in terms of relative root mean squared error (*ReRMSE*) (Breiman et al., 1984), of the ensembles learned using different methods. To properly assess the significance of the differences between these performances, we will follow a standard statistical procedure recommended by Demšar (Demšar, 2006). More specifically, we employ the corrected (Iman & Davenport, 1980) Friedman test (M. Friedman, 1940), followed by two post-hoc tests: the Nemenyi test (Nemenyi, 1963) and the Bonferroni-Dunn test (Dunn, 1961).

Perform extensive empirical analyses to obtain optimal design choices for each of the proposed methods for constructing ensembles of process-based models

In order to obtain the best performance of the proposed ensemble methods, we will perform extensive analyses in order to seek for the their optimal design properties. More specifically, we will first investigate what is the best approach to selecting the ensemble constituents. This includes testing whether it is better to fill the ensemble constituent set with process-based models with good performance measured on the training dataset or on a separate validation set. Next we will analyze how many constituents are sufficient for optimal performance. Finally, we intend to evaluate which of the proposed combination schemes yields better predictive performance.

Conduct a thorough and fair empirical evaluation of the predictive performance of the different proposed ensemble methods

The main contribution of this thesis is the novel methodology for learning ensembles of process-based models. In this context, we aim at fair and thorough empirical evaluation of the proposed methodology which will adequately address the first two hypotheses we conjectured previously. To this end, our first objective will be validating the output of the proposed ensemble methods in terms of their reasonability when employed in the context of modeling dynamic systems. This will include visual inspection of the obtained trajectories addressing the question of whether they resemble the measured behavior in the modeled system.

The second objective will consider comparison of the long-term predictive performance obtained by the proposed ensembles to the state-of-the art single process-based models. More specifically, our aim here is two-fold. On one hand, we address the question of whether this ensemble methodology, in general, can yield improved predictive performance when compared to a single model. In order to properly address this, the empirical comparison will include comparing the performance of every proposed method for learning ensembles of PBMs to a single process-based model on predictive modeling tasks in real-world aquatic ecodomains.

On the other hand, we aim at identifying the most appropriate process-based ensemble method for the task of modeling dynamic systems. To this end, we will consider two criteria. The first criterion will consider the predictive performance of each ensemble method. The second criterion will address the computational complexity of different methods for constructing ensembles of PBMs. Here, besides measuring the time needed for learning an ensemble, we will also investigate the methods in terms of architecture (whether they can be optimized or parallelized) and applicability (whether they are flexible and general enough for tackling modeling tasks in different domains).

Measure the influence of the intra-ensemble diversity on the predictive performance of the ensemble

In machine learning there is no accepted theory which formalizes how ensembles should be constructed in order to constantly yield improved predictive performance. The general consensus (Kuncheva, 2014; Ho, 2000; Hansen & Salamon, 1990; Hastie & Tibshirani, 1990), though, is that an ensemble should be constructed with accurate and diverse base models. In terms of accuracy, the base models should at least perform better than random guessing.

In terms of diversity, the base models should be independent in making prediction, i.e., they should err differently when new (unseen) test data is presented to them. While, intuitively these two requirements are sufficient for a good-performing ensemble, the findings of Kuncheva and Whitaker (2003) show that there is not always a strong correlation between them and the question of accuracy–diversity trade-off is more problem/task dependent than general. In this context, we intend to investigate whether the diversity between ensemble constituents influences the overall performance of process-based ensemble models, thus addressing our last hypothesis.

1.3 Contributions

This thesis contributes to several important aspects of both classical ensemble learning, and more importantly, to the process-based modeling paradigm applied to to the task of predictive modeling of real-world dynamic systems. The novel methodology presented in this thesis is partially published in several journal publications (Simidjievski, Todorovski, & Džeroski, 2015a, 2015b, 2016). The complete list of papers related to this thesis is given in the chapter titled "Bibliography". The main contributions of the work presented in this thesis are summarized as follows:

Contribution 1 *A novel methodology for learning ensembles of process-based models.*

In this thesis we address the task of learning ensembles of process-based models by designing, implementing and evaluating the appropriate methodology. The developed methodology is general and modular which allows for adapting different ensemble methods to the particular context of learning process-based models. This methodology is the main contribution of this thesis, since it extends the scope of current process-based modeling approaches to the task of learning ensembles of process-based models. To this end, the proposed ensemble methodology consists of four different algorithms for learning homogeneous ensembles of process-based models, as well as an algorithm for simulating them.

This contribution is also important in the wider context of the ensemble learning paradigm, applied in the context of time-series predictive tasks. While such ensembles have a narrow focus on short-term prediction tasks, where the value of the time series at the next time point is predicted, ensembles of process-based models provide accurate long-term predictions over many future time points.

Contribution 2 *A general evaluation framework for evaluating the predictive performance of the ensembles of process-based models.*

The second contribution is the design and execution of the extensive empirical evaluation for estimating the predictive performance of the proposed ensemble methodology.

Moreover, it enables the identification of the optimal design properties of the different ensemble methods proposed in this thesis. To this end, the experimental evaluation involves a series of predictive tasks of modeling dynamics in real-world ecosystems, which allows for thorough and fair evaluation of the predictive performance of the proposed methods.

Note, that this contribution is also important in the wider context of the process-based modeling paradigm. While the state-of-the-art approaches focus mainly on measuring the descriptive performance of the obtained process-based models or perform a cross-validation, here we propose a standardized procedure for measuring performance of models when applied to tasks of modeling future behaviors, i.e., beyond the time-period captured in training data. Namely, we consider this as a tractable framework for formal analysis of methods developed and applied within the process-based modeling paradigm.

Contribution 3 *Improvement of the long-term predictive performance over the state-of-the-art approaches to process-based modeling of dynamic systems.*

In this context, we compare the predictive performance of different ensembles of process-based models to the performance of a single process-based model. We conduct the empirical evaluation on the task of modeling population dynamics in aquatic ecosystems. The case studies considered concern modeling phytoplankton growth, a complex non-linear dynamic process, in three different real-life aquatic ecosystem domains. Based on the performed empirical evaluation, we also identified the main design decisions that need to be made when learning ensembles of process-based models. In general, all proposed ensemble methods, following the respective designs, improve the predictive performance of single process-based models. More importantly, when learning ensembles of process-based models by bagging and random library subsamples methods, this performance improvement is statistically significant.

This contribution is also important in the wider context of ensembles for time-series forecasting. While forecasting ensembles have a narrow focus on short-term prediction tasks, where the value of the time series at the next time point is predicted, ensembles of process-based models provide accurate long-term predictions over many future time points.

Contribution 4 *Case studies of predictive modeling of population dynamics in aquatic ecosystems.*

Finally, the thesis contributes to the realm of ecological modeling. The results of the performed experimental evaluation confirm our first two hypotheses that ensembles of process-based models, in general, provide logically plausible and, even more, accurate predictions of concentrations of species in an aquatic ecosystems than a single process-based model. This is a significant improvement in predictive performance over the state-of-the-art models of population dynamics, which, while focusing on providing an accurate explanation of the behavior of the observed system, struggle to achieve a satisfactory performance at predicting population dynamics over long periods (Atanasova, Todorovski, Džeroski, Remec, et al., 2006; Atanasova, Recknagel, et al., 2006). Note, however, while the proposed methodology has been only used in the limited context in the domain of ecological modeling, it is easily applicable and extendible towards other application domains (such as other aquatic domains, systems neuroscience and systems biology).

1.4 Organization

In order to improve the readability, this thesis is organized in four parts : *Introduction & Background*; *Methodology & Experimental Design*; *Results* and *Conclusion.* More specifically, the manuscript is structured as follows:

Part I *Introduction & Background*

This chapter, Chapter 1, introduces the research questions and the topics of interests presented in this thesis. It also provides the motivation for performing the presented research and sets it in the context of formulated hypotheses and specific set of goals for addressing them. Finally, it summarizes the main original scientific contributions.

Chapter 2 presents a relevant background to the work in this thesis. In the first part of the chapter we first give an overview of the process-based modeling paradigm in terms of related work, and then describe the task of learning process-based models. In the second part, we mainly focus on the ensemble learning paradigm. We first give an outline of the related work in this field and define the key ingredients for learning ensemble models in general.

Part II *Methodology & Experimental Design*

Chapter 3 gives a detailed description of the developed methods for learning and simulating ensembles of process-based models, which are the main contribution of this thesis. We first investigate the algorithm for simulating ensembles of process-based models. Next, we provide the necessary details for the four ensemble methods that combine process-based models in the constituent set. Finally, we analyze the computational complexity for learning ensembles with the proposed methods.

In Chapter 4, we present the experimental setup of the empirical framework for evaluating the developed ensemble methods on five tasks of modeling population dynamics in aquatic ecosystems. More specifically, we test the utility of ensembles of process-based models in the context of modeling three real-life ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan, and Lake Zurich in Switzerland. Additionally, we describe the necessary settings of the algorithms employed and give an overview of the performance metrics and statistical test used for assessing the performance of the ensembles.

Part III *Results*

In this part we present the results of the empirical analyses of the methodology for learning ensembles of process-based models, which serve both to find the optimal design choices and to illustrate the utility of the different algorithms for learning ensembles of process-based models. Chapter 5 presents the results of the empirical evaluation of ensembles of process-based models learned by sampling data instances. Next, Chapter 6 depicts the results of the analyses of ensembles learned with random sampling of the library of domain knowledge. The results of the evaluation of the method for learning ensembles with bagging of random library subsamples are presented in Chapter 7. Finally, the findings about the influence of intra-ensemble diversity on the predictive performance of the ensembles of process-based models are also presented in Chapter 7.

Note that some of the results in this thesis were already published in peer-reviewed journals, hence we present them as they appeared in the respective journal.

Part IV *Conclusions*

Part IV consists of a single Chapter 8 that concludes the thesis. It presents an overall summary of the thesis, summarizes the findings of the experiments and discusses them in the context of related research. Moreover, it lists the original contributions and gives directions for further work.

Chapter 2

Background

The research presented in this thesis lies in the intersection between two well established paradigms of machine learning, i.e. process-based modeling and ensemble learning. In this chapter we give a brief synopsis of both of these paradigms, which presents a relevant background to the work in this thesis. In the first part of this chapter, we first give an overview of the process-based modeling paradigm in terms of related work, then describe the task of learning process-based models. In the second part, we mainly focus on the ensemble learning paradigm. We first give an outline of the related work in this field and define the key ingredients for learning ensemble models in general.

Note that the research related to both process-based modeling and (especially) ensemble learning is extensive and present in a plethora of different scientific domains. It is beyond the scope of this thesis to give a complete and extensive overview on all related work, therefore we focus here on the most necessary details of both paradigms that are the basis of the work presented in this thesis.

2.1 Process-Based Modeling

Mathematical modeling is a widely used formalism in science and engineering employed for representation and reasoning about the complex reality. Scientists and engineers construct complex mathematical models with the purpose to study, control or predict the behavior of real-world systems under various conditions. In turn, they perform a variety of numerical and theoretical analyses to these models, aiming to reveal important insights into the system being observed and its domain in general. Constructing mathematical models however, is a nontrivial and demanding task, which involves an extensive expert knowledge and empirical observations (measurements) of the system at hand.

2.1.1 Modeling dynamic systems – concepts

In essence, mathematical models are complex rules which provide mapping of the *inputs/signals*, derived from knowledge and measurements of the system, into *outputs/feedbacks* which quantitatively represent the resulting (desired) behavior of the system. Formally, these complex rules are represented as a set of *equations* which are comprised of *variables* (the internal state and the input of the system) and the *constants/parameters* related to the components of the modeled system.

In the process of constructing a mathematical model of a real-world system, the first milestone is determining the underlying modeling formalism. The formalism, however, can vary based on the nature of the system and how the system is being modeled. At top-level we can distinguish between *static models* and *dynamic models*. The former class, also referred to as steady-state models, assumes that all variables in the system are constant. In contrast, dynamic models are employed for representing continuously evolving *dynamic systems.* This modeling framework can be further refined based on the amount of states the system can change into. Here we can distinguish between *discrete* and *continuous* models. Discrete models are comprised of difference equation, employed for modeling systems with finite (countable) number of states, as opposed to continuous models which comprise differential equations and focus on modeling continuous changes of the system at hand.

Ordinary Differential Equations (ODEs) are one of the most widely used mathematical formalism for modeling continuous dynamic systems. Here a model is represented as a set of differential functions, each of them consisting of one state variables and its derivatives. ODEs, however, are used in the limited context of modeling continuously evolving dynamic systems over one dimension (typically temporal). *Partially Differential Equations (PDEs)*, extend this scope to modeling changes of a continuous dynamic system over potentially infinite dimensions (typically two – spacial and temporal).

Note that, both modeling formalisms of ODEs and PDEs are purely *deterministic*, i.e. such models always produce an exact output of the behavior of the system uniquely determined by the initial conditions of the model. In reality however, the behavior of many systems is uncertain. These systems are modeled with *stochastic models*, which employ either probability density functions or stochastic differential equations to account for the uncertainty of the modeled behavior.

In this thesis, our aim is modeling continuous dynamic systems. More specifically, we focus on constructing deterministic dynamic models which change over time, thus the modeling formalism employed is ODEs. In this context, we can formalize a temporal change of a set of system variables X (also referred to as *endogenous* variables), as a function of its derivatives x and a set of forcing variables E (also referred to as *exogenous* variables). The exogenous variables are input/independents variables which drive the dynamics of the system, whereas the endogenous variables are dependent and formalize the behavior of the system being modeled. Typically, an ODE model of dynamic system has the form:

$$\frac{dX}{dt} = F(X(t), E(t), I, P),$$

where $\frac{dX}{dt}$ denotes time derivatives of the endogenous variables X over time t. In the function F, that denotes the structure of the ODE model, E represents the exogenous variables, I denotes a set of initial values of the exogenous variables (the values of the variables X in time t_0) and P represents the set of constant parameters of the model. Such a structure of ODEs with proper parameters, given initial values of the endogenous variables and the exogenous variables, can be employed for long-term simulation of the dynamic system's behavior, also known as *initial-value problem.*

Simulation of ODEs is an integration problem which relies on numerical approximation methods for nonlinear stiff (that consist of variables with both rapid and slow dynamics) ODEs. This however is a non-trivial and very time consuming task. There are several approaches which tackle numerical integration such as teacher-forcing simulation (Williams & Zipser, 1989) and full simulation (Gershenfeld, 1999). Full simulation, employed in this thesis, refers to a standard approach for solving numerical integration problems using only the initial state and input variables of the system. The derivatives of the state variables in every time point are approximated using changes in one (Euler or Runge-Kuta methods) or several (Adams-Moulton methods) of the previous time points. An example of full simulation, performed over a period of approximately one year, is presented in Figure 2.3 in Section 2.1.4.

Once a modeling framework has been determined, the next step in establishing an acceptable mathematical model is the procedure of *model construction*. Procedurally, model construction relates to two tasks of: (1) inferring the proper structure of equations – *structure identification task* and (2) determining acceptable values of parameters in the equations – *parameter estimation task*. The procedure of model construction, however, is typically an ill-posed problem and thus is often performed in conjunction with *model validation*. Model validation is the procedure of assessing the performance of the model. The procedures of model construction and model validation both relay on the measured data and/or the domain knowledge provided at input. While the model construction procedure uses this input to properly determine the structure and approximate the parameters of the model, the model validation procedure evaluates the goodness of the model both how it quantitatively matches the measured data and how it qualitatively resembles the provided knowledge.

2.1.2 Automated modeling of dynamic systems

In traditional system modeling, both model construction and model validation are performed by a (human) domain expert. However, there are a variety of more sophisticated methods, developed in different research areas, that employ automated approaches for establishing models of dynamic systems from knowledge and measured data. These methods mostly differ in the amount of knowledge and data used in the process of model construction, i.e. structure identification and parameter estimation. Based on the amount (and type) of knowledge used in generating models of dynamic systems, we can lay out a spectrum of automated modeling methods. On one end of this spectrum, we have purely (1) empirical (also referred to as data-driven or statistical) modeling methods which use only measurements in the process of establishing a model. On the other end, we have (2) theoretical (also referred to as knowledge-driven or mechanistic) modeling methods which aim at constructing understandable models using domain knowledge.

Considering data-driven approaches to modeling dynamic systems, research in the area of system identification (Ljung, 1999) mostly focuses on developing purely empirical methods for solving the task of parameter estimation. These methods rely on a data-driven trial-and-error cycle of trying out alternative structures, for each of them performing parameter estimation which relies on the model fit against measured data for the provided model structure. These structures typically belong to a well-defined class of models, such as linear regression, neural networks, fuzzy models and trees.

In the context of mathematical modeling of linear dynamic systems, typical system identification approaches relate to time-series regression methods such as linear regression, autoregressive-integrated-moving-average (ARIMA) modeling (Box, Jenkins, & Reinsel, 1994) or finite impulse response (FIR) system modeling. On the other hand, for modeling non-linear dynamics there are two approaches. The former considers techniques of linearizing the modeling problem before tackling it with standard techniques for modeling linear dynamic system. The latter, i.e. non-linear system identification (Nedellec, Rouveirol, Ade, Bergadano, & Tausend, 1996), includes more sophisticated techniques such as neural networks, fuzzy models and fuzzy linear model trees (Aleksovski et al., 2015). Note that the area of system identification also extends towards incorporating knowledge into the process of constructing models, where the structure identification problem is solved either by a human expert or provided at input based on a minimal a priori knowledge about the processes involved in the system. Such an approach is employed by Barzel, Liu, and Barabási (2015), where the authors reconstruct minimal models of microscopic dynamics in complex dynamic networks by employing background knowledge, subsequently estimating the constant parameters using empirical data.

Within the area of evolutionary computing, the approach of genetic programming (Koza, 1992, 1994) has been successfully applied to the task of identifying non-linear algebraic structures of dynamic systems from observed data (Gray, Murray-Smith, Li, Sharman, & Weinbrenner, 1998). Schmidt and Lipson (2009) use a similar approach to discover physical laws from measured data without employing background knowledge, where the task of structure identification relates to symbolic regression which relies on combining different segments of mathematical functions and operators using genetic programming. In a similar context, Schmidt et al. (2011) employ symbolic regression based on genetic programming for automated construction of metabolic networks. These methods suffer the same shortcomings as any data-driven approach to modeling, i.e., they are limited by the amount of knowledge provided at input. Given that the model validation is performed quantitatively, the inferred models can often be implausible. To address this limitation, Daniels and Nemenman (2015) induce phenomenological models by employing Bayesian Information Criterion for selecting a models from a hierarchy of models that captures typical behavior of the modeled system.

On the other end of the spectrum, within the domain of artificial intelligence, the area of qualitative reasoning (QR) (Kuipers, 1994) addresses the task of establishing qualitative representations of dynamic systems and reasoning about them. Qualitative Process Theory (QPT) formulates a framework for constructing qualitative models (Forbus, 1984), where a model relates to a physical process in the system at hand. Such models are specified by the components and parameters in the system which interact with the process and internal/external activities that affect the process. This framework is further extended in the automated approach of compositional modeling (Falkenhainer & Forbus, 1991) which relies on formalizing the conceptual knowledge of the system into libraries of model fragments that specify the system structure, its states and state transitions, causality and behavior. In turn, compositional modeling methods combine these modeling fragments into a qualitative model which most adequately resembles the system at hand.

In context of mathematical modeling, the QPC (Farquhar, 1993) automated modeling framework follows the compositional modeling paradigm and additionally employs Qualitative Differential Equations (QDEs), an abstract representation of ODEs used as mathematical qualitative modeling formalism, which can be simulated using the QSIM simulator (Kuipers, 1994). Another notable method, in this class of modeling approaches, is the PRET reasoning system (Bradley, Easley, & Stolle, 2001) which tackles the task of automated modeling of dynamic systems by incorporating a two-level structure of modeling knowledge. PRET uses the first level for inducing the model structures from the observed behavior, subsequently employing the second level of knowledge based on ODEs, which constraint the space of plausible model candidates. Recently, the Garp3 (Bredeweg, Linnebank, Bouwer, & Liem, 2009) offers a graphical interface for generating and simulating qualitative models represented as graphs, useful in situations when measured data is not available.

Different approaches to modeling, however, have direct influence on the resulting model. Models obtained with the data-driven approaches are often purely quantitative and considered as *black-box* models: While they are being easily constructible, require less recourses and are very accurate at predicting the systems' behavior, they do not necessarily reveal qualitative information about the interactions in the system in general. On the other hand, the knowledge-driven models have qualitative properties, thus are considered as *white-box*, i.e., they are comprehensible enough and allow for better understanding of the system's structure and performance. However, their construction demands an extensive knowledge about the system at hand and is very time-demanding especially for modeling complex dynamic systems.

Equation discovery (Langley et al., 1987), an area within machine learning, joins the two ends of the modeling spectra by studying methods for learning both model structure and parameter values from knowledge and observations (Džeroski & Todorovski, 1993; Todorovski & Džeroski, 2007). Early approaches to equation discovery strongly relate to system identification methods, however differ at a very important aspect. Most system identification methods emphasize the task of parameter estimation and circumvent the task of structure identification when modeling a real system. Conversely, equation discovery methods besides tackling the task of parameter estimation, they strongly focus on identifying the mechanistic structure of the modeled system from observations. In the development of equation discovery methods through the years, the focus has shifted towards incorporating knowledge into the process of establishing a model. In the next section, we will review the development of the equation discovery paradigm from the first proposed approach for equation rediscovery till the state-of-the art process-based modeling paradigm employed in this thesis.

2.1.3 From equation rediscovery to process-based modeling

The early approaches to equation discovery mainly focused on rediscovering scientific laws of physical systems from observations. The results of preliminary experiments have indicated that many complex processes, previously described in the literature, can be (re)discovered/reconstructed by employing machine learning algorithms. The pioneering method in the area of equation discovery, BACON (Langley et al., 1987), incorporates a set of data-driven heuristics for discovering patterns (trends) in the measured data, subsequently formulating hypotheses about them. It starts by exhaustively generating hypotheses involving every two variables in the system, subsequently estimating parameters of each of these interactions (much like in system identification). The several successful applications of this method (Langley et al., 1987) set in motion the research in machine learning for discovering quantitative/qualitative laws of real-world systems. Modeling of real-world systems, however, requires utilizing background knowledge in the process of modeling. To this end, equation discovery methods incorporate background knowledge in the learning process (Lavrač & Džeroski, 1994) as an inductive bias (Nedellec et al., 1996).

Inductive bias refers to any set of assumptions used by the learning algorithm for obtaining a (general enough) output from a specific input. Inductive machine learning algorithms allow for different types of inductive bias such as: language bias (constraints of the hypotheses space), search bias (defining what part of the hypotheses space is being searched) and validation bias (defining stopping criteria). The majority of the developed equation discovery methods incorporate language bias in the learning process, which is used to constraint the space of plausible candidate modeling components that can be combined. Here additionally we can distinguish between different equation discovery methods based on how the language bias is specified. This includes methods that employ: non-declarative (fixed space of modeling components, determined by parameters specified at input), parametrized (fixed space of modeling components, determined by parameters specified by the user), or declarative (user customized space of modeling components based on background knowledge) language bias.

E* (Schaffer, 1993) is one of the earliest equation discovery methods which incorporates non-declarative language bias. It uses a reasonably small set of predefined class of equation components which relate to only two variables. More sophisticated methods including COPER (Kokar, 1986), ABACUS (Falkenhainer & Michalski, 1986), EF (Zembowicz & Zytkow, 1992), LAGRANGE (Džeroski & Todorovski, 1995) and SDS (Washio & Motoda, 1997), extend this scope towards different equation structures such as polynomials and trigonometric functions. Additionally they allow for some degree of user intervention in the learning process such as degree of induced polynomials or number of induced equation terms. These methods however, have two major limitations: they use limited (domain-independent) background knowledge and are employed primarily for inducing algebraic equations (except LAGRANGE) not suitable for modeling dynamic systems.

Considering the former limitation, the background knowledge usually includes information about the measurements and units of the observed variables. Considering the latter, even though LAGRANGE has the ability to induce components in the form of ordinary differential equations, the derivatives are computed beforehand and such differentiation introduces a large numerical complexity. GOLDHORN (Križman, Džeroski, & Kompare, 1995) is an extension of LAGRANGE which addresses the numerical overhead by substituting the differentiation with numerical integration thus improving the overall performance of the algorithm.

To address both of the afore-mentioned limitations, Todorovski and Dzeroski proposed LAGRAMGE (Todorovski & Džeroski, 1997), which employs sophisticated declarative language bias in a form of context-free grammars for defining the space of possible equation structures. LAGRAMGE also allows parametrization in terms of stopping criteria and search heuristics. This method is capable of discovering ODEs using the same method for numerical integration previously proposed in GOLDOHORN. While the grammars are a very elegant solution for expressing a variety of different domain-specific components, they are fairly complex for construction and are task specific (grammar constructed for one modeling task cannot be reused for another). Additionally, several experiments proved that context-free grammars are not the most suitable formalism for encoding background knowledge, in terms of comprehensibility, when used by a domain expert.

Process-based modeling (PBM), refers to a modeling paradigm within equation discovery, which includes a subset of methods that combine both the empirical and theoretical approaches to modeling. These PBM methods employ process-based domain-specific background knowledge as declarative language bias, that in conjunction with observed data, can address the task of modeling real-world dynamic systems both in terms of structure identification and parameter estimation. LAGRAMGE2.0 (Todorovski, 2003; Todorovski & Džeroski, 2007) is the pioneering PBM method, which is an extension to LAGRAMGE in terms of process-based modeling knowledge. The background knowledge is organized around the central notion of entities and processes (similar as in QPT), which in turn is translated to context-dependent grammar. LAGRAMGE2.0 uses LAGRAMGE as an underlying method to heuristically induce candidate models and fit their parameters using the measured data. However, LAGRAMGE2.0 has the same limitation as its predecessor, which relates to the complexity of the modeling formalism and the applicability to different modeling tasks. CIPER (Pečkov, Džeroski, & Todorovski, 2008) employs background knowledge in a form of subsumption constraints to heuristically search for polynomial algebraic equations, however it does not support discovery of ordinary differential equations.

Inductive Process Modeling (IPM) refers to a method proposed by Bridewell et al. (2008), which formalizes the background knowledge as a process-based library where the components are specified as generic processes. These generic processes serve as general placeholders of different variables and constants together with their constraints which relate to the modeling task at hand, such as types and value ranges. In turn, by employing heuristic search for instantiating these generic process into specific ones and combining them, IPM constructs process-based models (PBMs) – an accurate, understandable and modular representation of the observed system.

Process-based models have several characteristics which make them very efficient for modeling dynamic systems. First, they provide a conceptual representation of the structure of the modeled system, depicting the high-level relations (processes) between the system components (entities). Second, this high-level process-based representation is translated into a low-level mathematical formalism depicted as a set of differential and/or algebraic equations, which in turn are employed for simulation of the systems behavior. Finally, the library of domain-specific knowledge allows for instantiation of a number of different building blocks for generating process-based models, which is particularly relevant for algorithms tackling the task of automated learning of process-based models from data.

Procedurally, IPM facilitates the additive combination of any set of generic processes into a model structure. This, however, can lead to large unconstrained space of possible model structures, which first can limit the method in terms of increased computational overhead and second can result into logically implausible models. The Hierarchical IPM (Todorovski et al., 2005) addresses these limitations by organizing the background knowledge into hierarchies of generic processes and generic entities. This hierarchical structure allows for the inducing method to search through reduced space of model components, subserviently resulting in more reasonable model structures. SCIPM (Bridewell & Langley, 2010), on the other hand, tackles the model enumeration task by using constraint satisfaction methods which relate to explicitly encoded constraints. Finally, Bridewell et al. (2005) propose FUSE, an extension to HIPM, which improves the generalization ability of the process-based models by combining them into an ensemble. We further discuss the relation of FUSE to our approach to learning ensembles of process-based models in Section 2.2.5.

The utility of the process-based modeling approach has been shown in a variety of modeling tasks in different domains including modeling atmospheric cycles (Todorovski, 2003), ecological modeling (Džeroski & Todorovski, 2003; Atanasova, Todorovski, Džeroski, Remec, et al., 2006; Atanasova, Recknagel, et al., 2006; Bridewell et al., 2008; Simidjievski et al., 2015a), mechanics and hydrodynamics (Džeroski & Todorovski, 1995; Bridewell et al., 2008), and systems biology (Langley, Shiran, Shrager, Todorovski, & Pohorille, 2004; Taškova et al., 2011; Tanevski et al., 2015). The state-of-the-art implementation of the process-based modeling paradigm is the Process-Based Modeling Tool (ProBMoT) (Čerepnalkoski, 2013), a software platform for complete modeling, parameter estimation, and simulation of process-based models. It extends HIPM with explicit constraints (assumptions) for a particular domain at hand and employs a variety of meta-heuristic optimization methods. In this thesis, we use ProBMoT [1] as the base learning algorithm for learning constituents of ensembles of process-based models. For that reason, ProBMoT will stand as a proxy for describing the most important details of the task for learning process-based models, presented in the next section.

2.1.4 Process-based models

The process-based modeling paradigm addresses the task of constructing process-based models of a dynamic system. In essence, process-based models provide a conceptualization of the structure of the observed system, accompanied by modeling details that allow for their transformation to equations and therefore simulation. More specifically, they tackle the task of representing dynamic systems from two aspects: qualitative and quantitative. From a qualitative aspect, a process-based model is a set of entities and processes. The entities represent the components of the observed system, which are involved in activities represented by the processes. From a quantitative aspect, a process-based model is interpreted as a set of ordinary differential and/or algebraic equations which are employed for

[1]Available at http://probmot.ijs.si

simulating the behavior of the observed dynamic system. In essence, process-based models encode both the low-level quantitative mathematical formalism and add a high-level qualitative description of the system.

To properly assess the relevant details of the process-based modeling paradigm, we are going to illustrate its use on a simple example of modeling population dynamics in an aquatic ecosystem. Models of aquatic ecosystems are required for better understanding, prediction and management of such systems (Jørgensen & Bendoricchio, 2001). These models target the relations between entities, i.e., nutrients, primary producers, animals and environmental changes that typically occur in aquatic ecosystems (Luenberger, 1979). Figure 2.1 depicts a cyclic relationship involving a primary producer (phytoplankton, abbrev. *phyto*) that grows by feeding on nutrients (nitrogen and phosphorous), the concentrations of which are influenced by the environment and the process of respiration (Atanasova, Todorovski, Džeroski, & Kompare, 2006).

In order to model such a system using the process-based paradigm modeling, we first need to formalize the modeling knowledge. Process-based modeling allows for a high-level representation of knowledge, cataloged in a domain-specific library of entity and process templates. The templates embody general properties of the interactions that govern the dynamics in the domain at hand and serve as recipes for establishing specific entities and processes observed in a given system.

Entities comprise variables and constants related to the components of the observed system. For example, an entity representing phytoplankton in an aquatic ecosystem would include a variable corresponding to its concentration that changes through time, and a constant corresponding to its maximal growth rate. Each entity variable has three important properties: the role in the model, the initial value and the aggregation function. The role of the variable in the model can be endogenous, i.e., representing internal system state, or exogenous, i.e., representing an input external to the system (not modeled within the system). An example of an endogenous variable in an aquatic ecosystem is the concentration of phytoplankton, while the environmental temperature is often treated as exogenous. Initial values of endogenous variables are necessary for model simulation. Moreover, each endogenous variable has its constraints defined, which limit the set of feasible values of the variable (for example, the concentration of the phytoplankton cannot be negative nor can exceed 100 gWM/m^3). Finally the aggregation function for a variable specifies how influences from multiple processes on the specific variable are need to be combined, e.g., additively or multiplicatively.

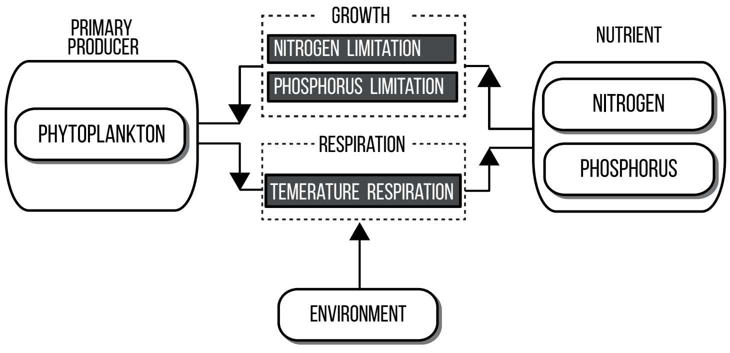


Figure 2.1: Graphical representation of the relations (arrows and black boxes) between the entities (oval transparent boxes) in a simple lake ecosystem.

The processes include specifications of the entities that interact, equations, and sub-processes. Consider the process of growth. It involves the phytoplankton as well as the growth limiting factors of nutrients and the environment. Equations provide the model of the interaction represented by the process and contains variables and constants from the entities involved in the corresponding interaction. In the phytoplankton growth example, an equation would define the mathematical model for calculating the growth rate. Finally, each process can include a number of sub-processes related to different aspects of the interaction. For example, the term of nutrient limitation of growth (or nitrogen/nutrient) can be specified in an appropriate nitrogen/phosphorous limitation sub-process of the growth process. Sub-processes improve both the interpretability and the modularity of process-based models (Bridewell et al., 2008).

Table 2.1 depicts an example of a simple library for modeling population dynamics in aquatic ecosystems. The first four declarations correspond to template entities organized in a hierarchy. The template entities of ***EcosystemEntity*** and ***Environment*** are at the top of the hierarchy; the first corresponds to the entities of the aquatic ecosystem, while the second to its environment. Down the hierarchy, the ***EcosystemEntity*** is then specialized into the two template entities of ***PrimaryProducer*** and ***Nutrient***.

Each entity template may include constant and variable properties, which are inherited down the hierarchy. The variable properties (denoted `vars`) are those which change over time: The ***EcosystemEntity*** has as variable property specifying its current concentration (denoted *conc*). Similarly, the ***Environment*** entity includes the ***temperature*** variable. The constant properties that do not change are denoted with `consts`, e.g., the template entity ***PrimaryProducer*** has the constant property ***maxGrowthRate***. Finally, note the specification of an aggregation function for each variable (denoted `aggregation`) which specifies that the influences on the ***conc*** variable of ***EcosystemEntity*** are summed up, while the influences on the ***nutrientLim*** variable are multiplied.

The template processes are also organized in a hierarchy and specify which entities can interact and how these interactions govern the dynamics of entity variables. Highest in the hierarchy of this aquatic ecosystem are the template processes of GROWTH and RESPIRATION. The RESPIRATION template process specifies the influence of the temperature in the process of respiration on the concentrations of the primary producer and the nutrients involved in the system. Note that the RESPIRATION template has two alternatives of LINEARTEMPRESPIRATION and EXPTEMPRESPIRATION which further specify whether the temperature influence is modeled linearly or exponentially, respectively. Similarly, the GROWTH template encodes the influences of growth on the same concentrations. Additionally, GROWTH involves a subprocess GROWTHRATE, which implies that a GROWTHRATE must be specified for each nutrient involved in the process of growth. The hierarchy also specifies two instances of the template process GROWHRATE, MONODGROWTHRATE and EXPSATURATEDGROWTHRATE, that correspond to two alternative models of growth limitation due to limited nutrient supply. Note therefore, the hierarchical structure of the process templates allows for the specification of modeling alternatives for an observed interaction between entities.

Given the library of model fragments (template entities and processes), we can now formulate the task of learning process-based models from knowledge and data as a search task. Namely, given the specific entities in the observed system at hand, one can instantiate the template processes from the library into a set of specific processes that can be considered for inclusion in the model of the observed system. In turn, based on this set of specific model components, we can specify the search space of combinations thereof. Some of the combinations can be rejected as implausible, due to further modeling assumptions made by the user, such as the presence or absence of certain processes in the model.

Table 2.1: Template entities and processes for modeling population dynamics in aquatic ecosystems. Here td(x) denotes the time derivative of x.

```
template entity EcosystemEntity{
    vars: conc {aggregation:sum;} }
template entity PrimaryProducer:EcosystemEntity{
    vars: nutrientLim{aggregation:product}, growthRate;
    consts: maxGrowthRate; }
template entity Nutrient:EcosystemEntity { }
template entity Environment { vars: temperature; }

template process GROWTH(pp: PrimaryProducer, ns: Nutrients) {
    processes: GROWTHRATE(pp, ns);
    equations:
        td(pp.conc) = pp.maxGrowthRate * pp.growthRate * pp.conc,
        td(ns.conc) = −n.alpha * pp.maxGrowthRate * pp.growthRate * pp.conc; }
template process GROWTHRATE(pp: PrimaryProducer, n: Nutrient) {}
template process MONODGROWTHRATE:GROWTHRATE {
    consts: halfSaturation;
    equations:
        pp.growthRate = n.conc/(n.conc + halfSaturation);}
template process EXPSATURATEDGROWTHRATE:GROWTHRATE {
    consts: saturationRate;
    equations:
        pp.growthRate = 1 − exp(−saturationRate * n.conc);}
template process RESPIRATION(
pp: PrimaryProducer, ns: Nutrients, env: Environment) {}
template process LINEARTEMPRESPIRATION:RESPIRATION {
    consts: respRate,refTemp,minTemp;
    equations:
        td(pp.conc) = −respRate * pp.conc * pp.conc
                      *(env.temperature − minTemp)/(refTemp − minTemp),
        td(ns.conc) = respRate * pp.conc * pp.conc
                      *(env.temperature − minTemp)/(refTemp − minTemp); }
template process EXPTEMPRESPIRATION:RESPIRATION {
    consts: respRate,refTemp,theta;
    equations:
        td(pp.conc) = −respRate * pp.conc * pp.conc
                      *pow(theta, env.temperature − refTemp),
        td(ns.conc) = respRate * pp.conc * pp.conc
                      *pow(theta, env.temperature − refTemp); }
```

Figure 2.2 represents the architecture of the ProBMoT software platform for process-based modeling. ProBMoT supports simulation, parameter estimation and automated learning of process-based models. The process-based learning algorithm, employed in ProBMoT, is outlined in Algorithm 2.1. It first takes as input *library* of domain-specific modeling knowledge, followed by *data* in the form of time-series measurements of the observed dynamic system. Since process-based modeling aims at modeling the behavior of a dynamic system over time, measured values are continuous, contiguous and may be non-uniformly

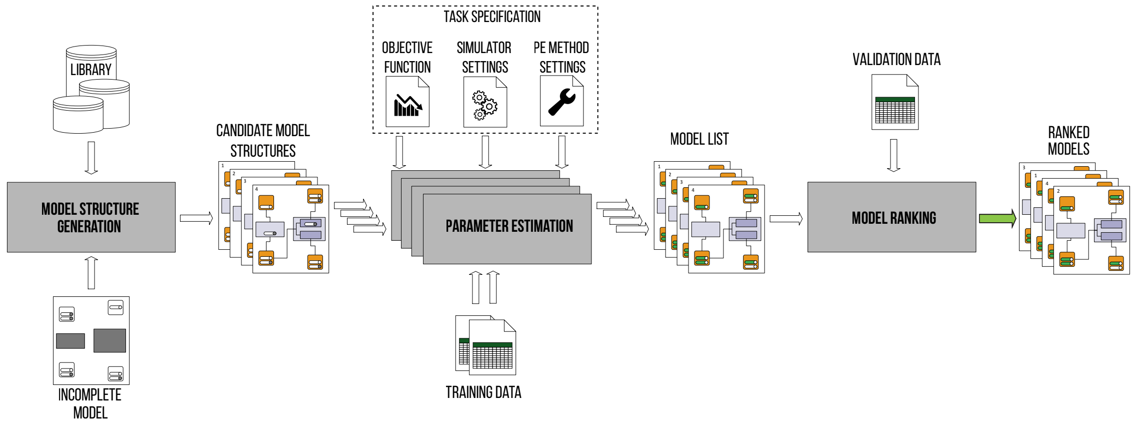


Figure 2.2: Graphical representation of the algorithm for learning process-based models from knowledge and data

distributed. The last input to the algorithm is an *incompleteModel* which specifies the expected logical structure of the expected model in terms of entities and processes observed in the system at hand that relate to the modeling assumptions made by the modeler.

First, the algorithm assembles all theoretically plausible model components by binding the entities of the observed system to the template processes from the library. Next, based on the incomplete model (taking into account the assumptions), the algorithm enumerates all the plausible candidate model structures. Each of these high-level structures is then compiled into a system of equations eligible for simulation. Before simulation, however, a parameter estimation task is being solved for the model structure at hand to obtain values of the model parameters that best fit the observed data.

To this end, the parameter estimation process is based on the meta-heuristic optimization framework jMetal 4.5 (Durillo & Nebro, 2011) that implements a number of global optimization algorithms. In particular, ProBMoT uses the Differential Evolution (DE) (Storn & Price, 1997) optimization algorithm. For simulation purposes, each process-based model is first transformed to a system of ODEs. ProBMoT allows for full simulation of dynamic systems, which is performed with the CVODE package. CVODE (C-package for Variable-Coefficient ODE), a solver from the SUNDIALS suite (Cohen & Hindmarsh, 1996), is a general-purpose ODE solver that uses linear multistep variable-coefficient methods for integration, i.e., Adams-Moulton method (for non-stiff problems) and backward differentiation formula (for stiff problems). More precisely, ProBMoT employs the Backward Differentia-

Algorithm 2.1: Outline of the generic algorithm for learning process-based models from knowledge and data.

```
  Input: $library, data, incompleteModel$
  Output: $modelList$
1 $components \leftarrow$ instantiate($library, incompleteModel$)
2 foreach $structure \in$ enunmerate($components, incompleteModel$) do
3     $modelEq \leftarrow$ compileToEquation($structure$)
4     $\{model, error\} \leftarrow$ parameterEstimation($modelEq, data$)
5     $modelList \leftarrow modelList \bigcup \{model, error\}$
  end
6 $modelList \leftarrow$rank($modelList, error$)
```

tion Formula (BDF) in combination with modified Newton's iterative method (for solving the non-linear equations at each step) and a preconditioned Krylov method (for solving the linear equations at each step) that is suitable enough for large stiff ODE models.

Moreover, ProBMoT implements a number of measures of model performance: the sum of square errors (SSE) between the simulated and observed behaviour, and several variants thereof. The latter include mean squared error (MSE), root mean squared error (RMSE), relative root mean squared error (ReRMSE) and weighted root mean squared error (WRMSE). After estimating the parameters for all candidate model structures, the algorithm outputs a sorted list of process-based models according to their validation error, i.e., the discrepancy between the model simulation and the observed system behavior.

After estimating the parameters for all candidate model structures, the algorithm outputs a sorted list of process-based models according to their error on training data, i.e., the discrepancy between the model simulation and the observed system behavior. Table 2.2 (top) presents a process-based model of the system depicted in Figure 2.1: note the one-to-one correspondence between entities and process depicted in the system graphical presentation and the process-based model. Each entity and process instance incorporate the variables and the constants related to the corresponding template.

Considering the mathematical formulation of the processes embodied in the library, the high-level representation of the interactions in the system is compiled into a system of algebraic and ordinary differential equations adequate for simulation. Table 2.2 (bottom) provides the quantitative formulation of the process-based model presented above, where $p_c(t)$, $ph_c(t)$ and $n_c(t)$ denote the concentrations of phytoplankton, phosphorous and nitrogen, respectively. The environmental temperature is denoted with T, and the growth rate of the phytoplankton with $gRate$. Since we are modeling only the concentration of phytoplankton (which is denoted as endogenous in Table 2.2 (top)), the system of equations consists of a single ordinary differential equation.

These equations can then be fully simulated, which results in a trajectory which is utilized for further analyses of the modeled system. Figure 2.3 presents the simulation of phytoplankton concentration obtained by using the learned process-based model. Here we used real data from Lake Bled (Atanasova, Todorovski, Džeroski, Remec, et al., 2006) for the exogenous variables involved in the system (nutrients and temperature) and for estimating the parameters of the system. The resulting simulation spans the time period of approximately one year, excluding the period when real measurements were not available due to freezing of the lake. The *DATA* trajectory (represented by a dashed line) represents real measurements that can be used for a visual assessment of the process-based model performance.

Table 2.2: A process-based model (top) of phytoplankton dynamics compiled to ODE (bottom), in the simple lake ecosystem from Figure 2.1, based on the template entities and processes from the library presented in Table 2.1.

```
//Entities
entity phyto : PrimaryProducer {
    vars: conc { role: endogenous; initial: 1.665;};
    consts: maxGrowthRate = 0.88; }
entity phos : Nutrient {
    vars: conc { role: exogenous;};
entity nitro : Nutrient {
    vars: conc { role: exogenous;};
entity env : Environment {
    vars: temp { role: exogenous;}; }

//Processes
process GROWTH(phyto, [phos, nitro]): GROWTH
    { processes: GROWTHRATE;}
process NITROGENLIM(phyto, nitro): EXPSATURATEDGROWTHRATE
    { consts: saturationRate =14.9;}
process PHOSOPHOROUSLIM(phyto, phos): EXPSATURATEDGROWTHRATE
    { consts: saturationRate =8.08;}
process RESPIRATION(phyto, [phos, nitro], env): LINEARTEMPRESPIRATION
    { consts: respRate=0.036, minTemp=0.542, refTemp=17.4;}
```

$$\frac{d\mathrm{p_c}}{d\mathrm{t}} = \mathrm{p_c(t)} \cdot \mathrm{gRate(t)} - 0.036 \cdot \frac{\mathrm{T(t)} - 0.542}{\mathrm{T(t)} - 17.4} \cdot \mathrm{p_c^2}$$

$$\mathrm{gRate(t)} = 0.88 \cdot (1 - \mathrm{e}^{-8.08 \cdot \mathrm{ph_c(t)}}) \cdot (1 - \mathrm{e}^{-14.9 \cdot \mathrm{n_c(t)}})$$

$$\mathrm{p_c(t_0)} = 1.665$$

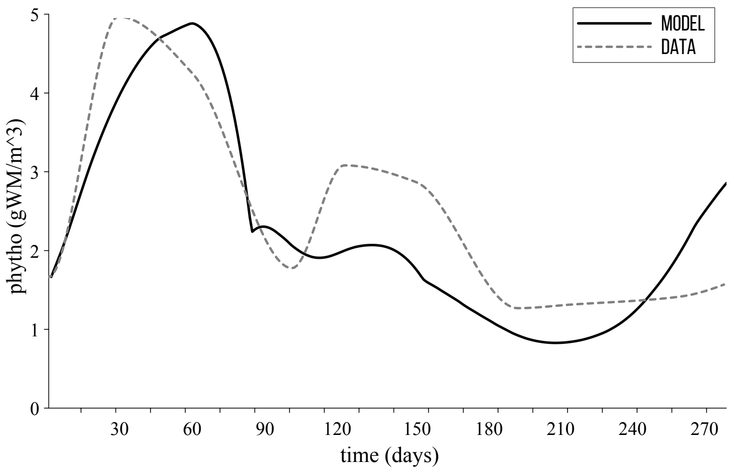


Figure 2.3: Simulation of phytoplankton concentration dynamics (solid line) as modeled with the process-based model from Table 2.2 and its comparison to observed phytoplankton concentration (dashed line).

2.2 Ensemble Learning

Ensembles are a well established machine learning paradigm, leading to accurate and robust models predominantly applied to predictive modeling tasks. Ensemble models comprise a finite set of diverse predictive models whose combined output is expected to yield an improved predictive performance as compared to an individual model.

In the literature, there is an ample amount of work dedicated to ensemble learning. This includes studying/developing of a variety of methods for constructing ensembles and application of ensembles to a plethora of predictive tasks (Okun, Valentini, & Re, 2011; Rokach, 2010). Considering the latter, in this section, we will focus on ensembles applied to the most common predictive tasks, i.e., the tasks of classification and regression, which will stand as a proxy for illustrating the concept behind ensembles in general. Regarding the former, several studies (Rokach, 2009; Kuncheva, 2014; Valentini & Masulli, 2002; Sharkey, 2002) focus on the architecture of an ensemble constructing method. Each of these studies proposes different taxonomies (at different levels and directions of detail) which serve as a blueprint for learning ensembles. However, the general consensus is that an ensemble method consists of two components: (1) technique/rule (referred to as a combining scheme or aggregation method) for combining the outputs of the multiple base models in a single prediction and (2) algorithm(s) for learning the ensemble constituents (referred to as base learners).

Considering how the constituents are combined into an ensemble output, there are two general approaches: model selection and models combination/fusion (Džeroski et al., 2009; Kuncheva, 2002). The former approach simply selects the best performing base model as the final ensemble output. The latter approach, incorporates different techniques for aggregating the outputs of the individual base models into an ensemble prediction.

The predictive power of ensembles, at least intuitively, lies in the diversity among the base models. While, in principle, any combined set of predictive models can be considered as an ensemble; in practice, more sophisticated ensemble learning methods are developed which tend to actively imply diversity in the constituents set that consequently improves the overall predictive accuracy. The second component of an ensemble method aims at designing/learning the base models. Here we can distinguish between homogeneous and heterogeneous techniques for constructing an ensemble. The former includes base models that are learned with the same learning algorithm, but from different samples of the training data. In the latter, the ensemble constituents are learned using different learning algorithms.

Today, ensemble learning and its applications hold a prominent place in the field of machine learning (Dietterich, 2000b; Kuncheva, 2014; Valentini, 2003; Seni & Elder, 2009; Rokach, 2010). However, it is nontrivial to trace-back to the origin of the ensembles, at least in the form that is employed nowadays. Probably, one of the most employed "many model" approaches is a key concept in the Monte Carlo methods, widely used in domains of life and earth sciences for stochastic simulations and numerical optimization.

In the context of learning theory, however, the concept of combining (and learning) multiple models, according to Kuncheva (2014), is first mentioned by Sebestyen (1962), where the author proposes a multi-level sequence of classification models, where the output of one model is used as an input for learning the next model in the sequence. In 1979, Dasarathy and Sheela (1979) proposed a two-model setup (one linear classifier and one kNN classifier), where the final prediction is selected from one of the models based on their accuracy in different partitions of the feature space. Zuev (1986) reported on learning probabilistic model of committee (ensemble) of classifiers. In the early 90's, several studies empirically showed that learning multiple models and combining their output can substantially reduce the predictive error as compared to learning a single model. These studies addressed a variety of predictive tasks by constructing ensembles using decision trees (Kwok & Carter, 1988), rules (P. B. Brazdil & Torgo, 1990; Kononenko & Kovačic, 1992), artificial networks (Hansen & Salamon, 1990; Baxt, 1992; Perrone & Cooper, 1993) for the task of classification, as well as regression tasks (Wolpert, 1992; Perrone, 1993).

In contrast, theoretically-wise, (Schapire, 1990) discussed the question of learnability and generalization power of a probably approximately correct (PAC) model. He suggested that instead of learning one (approximately) highly accurate model (ref. to as a strong learner), a *weak learner* (slightly better than random) can be learned and boosted to a better performance by several iterations of error penalization, subsequently performing equally or better than a single strong learner. The same year, Hansen and Salamon (1990), which focused more on the diversity between the base models, stated that if multiple N models have the sample probability of making errors and they err independently, the error of their combination will decrease monotonously as a function of N. These two studies together with the afore-mentioned empirical studies marked the conception of "modern" ensemble learning.

Since then, many studies have been published, both which empirically report on the predictive achievements of ensembles (Banfield et al., 2007; Bauer & Kohavi, 1999; Breiman, 1996a; Maclin & Opitz, 1999), as well as theoretically justifying their performance (Allwein, Schapire, & Singer, 2000; Breiman, 1996; Domingos, 2000; Geman, Bienenstock, & Doursat, 1992; Kong & Dietterich, 1995; Mason, Bartlett, & Baxter, 2000; Schapire, Freund, Bartlett, & Lee, 1997). Moreover, popular applications of ensembles include the winning solutions of the Netflix Competition (Bell, Koren, & Chris, 2008), The Higgs Boson Challenge (Melis, 2014), the Dream Challenges (Huynh-Thu, Irrthum, Wehenkel, & Geurts, 2010) etc.

In the remainder of the section, in Section 2.2.1, we will first focus on methods for constructing ensembles, followed by methods for learning the ensemble constituents in Section 2.2.2. Next, in Section 2.2.3, we will shift our focus towards techniques for obtaining interpretable ensembles. In Section 2.2.4, we will discuss the question of the performance of the ensembles in general, more specifically why do ensembles have good predictive performance. Finally, in Section 2.2.5 we are going to discuss the ensemble learning paradigm in the context of modeling dynamic systems in terms of related approaches to learning such ensembles.

2.2.1 Ensemble combining schemes

The first milestone when constructing ensembles is choosing the most appropriate technique for combining predictive models. As we mentioned previously, there are two general approaches for generating the output of the ensemble: model combination (or fusion) and model selection (Džeroski et al., 2009; Kuncheva, 2002). In the latter approach, each base model is first evaluated and the prediction of the best performing one is used as the overall ensemble prediction. The ensembles combined with this approach are better interpretable and usually computationally efficient for learning, however their predictive performance is limited by the performance of the selected model. In contrast, the former and more sophisticated approach, uses different combining schemes for aggregating the predictions of all ensemble constituents into an ensemble prediction. Here we can distinguish between (1)dynamic and (2)fixed combination schemes.

Dynamic combining schemes involve learning at different stages in the process of creating the ensemble; either learning how the base models are to be combined or even learning the base models. Stack generalization or simply — stacking (Wolpert, 1992), depicted in Figure 2.4, is the most popular ensemble method which includes dynamic combining scheme. Here, a predictive meta-model is learned from the combined predictions of the individual ensemble constituents. The ensemble constituents are first learned from different parts of the training data (and evaluated on the remaining parts) with (mostly often) different learning algorithms. In turn, their predictions are combined and used as an input to a new meta-learning algorithm. The prediction of the resulting meta-model is considered as the prediction of the ensemble. Stacking has been shown to be an efficient algorithm, however it is limited to the performance of the algorithm chosen for learning the meta-model.

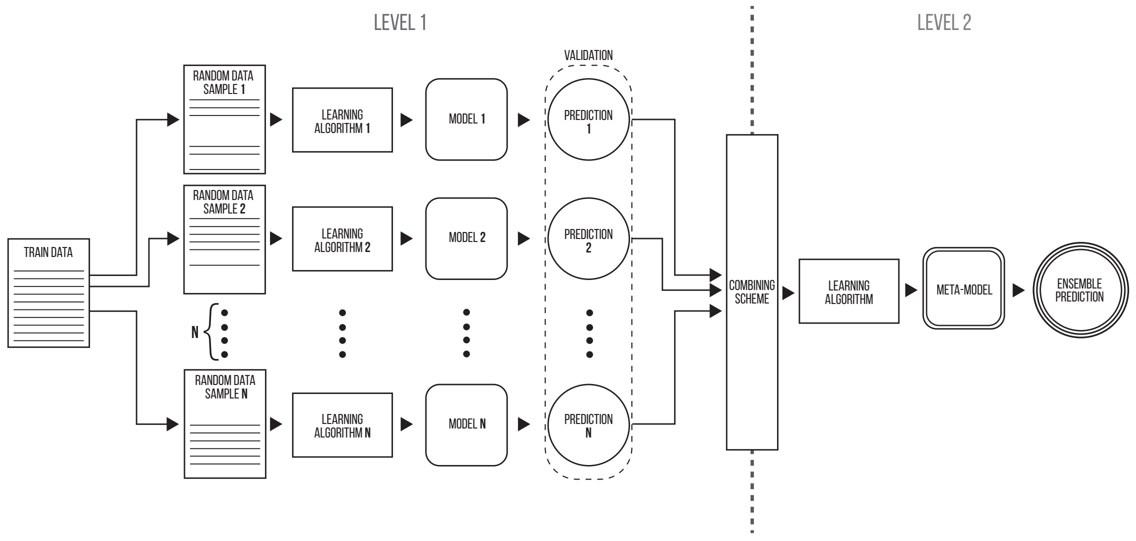


Figure 2.4: Illustration of the Stack generalization (Stacking) method for creating heterogeneous ensembles.

Another example of such scheme is employed in the so-called "architecture of classifiers" type of ensembles (Sebestyen, 1962; Ianakiev & Govindaraju, 2000). Such ensemble methods involve a daisy chain of predictive models, where each base model is learned using the prediction of its predecessor as a training dataset. The final ensemble prediction is then either taken from the last predictive model in the chain or all the predictions from the base models in the process are once more combined at the end. This method can also result in interpretable ensemble, however it is prone to overfitting and can be computationally very expensive. The final example of dynamic combining includes fusing structure fragments from the base models instead of their predictions (Bridewell et al., 2005). The resulting ensemble is fairly robust and interpretable, however this method usually involves additional user intervention and can be used with a limited set of types of predictive models.

In practice, though, the majority of ensemble methods employ a fixed (a posteriori) combining scheme. This involves first learning the base models, and then employing some combination rule for aggregating the individual predictions. There is a range of different fixed combining schemes varying based on the predictive task at hand and the desired output of the ensemble. In this section we will focus on the most popular ones which address classification and regression tasks: *voting* and *averaging*. For classification task, the most straightforward approach is uniform (majority) voting. Here, each base model first predicts a class value, and the class predicted by the majority of the base models becomes the output of the ensemble. Another alternative to voting is probability distribution voting (Kononenko & Kovačic, 1992). Each base model, instead of predicting the class value as in the previous case, predicts the probability of an example being in each class. The class that has the highest sum of probabilities is considered the output of the ensemble. For both alternatives of voting, there are also weighted variants. While in the uniform variants the vote of each base model is equal, here a weight is assigned to the constituents based on their performance (eg., accuracy, F-measure or even more complex measurements (Kuncheva, 2014)). More complex voting alternatives include Likelihood Combination (Ali & Pazzani, 1996), Bayesian Combination (Buntane, 1990; Ali & Pazzani, 1996; Domingos & Pazzani, 1997), probabilistic approximation (Kuncheva, 2014), singular value decomposition (Merz, 1999) etc.

For the task of predicting numeric values, i.e. regression, the most widely used (and simplest) fixed scheme is averaging: The mean value of every base-model prediction represents the overall ensemble output. Similarly to classification, a weighted variant can be used, where the assigned weights correspond to the performance (MSE, RMSE, ReRMSE, confidence) of the base models (Drucker, 1997). More complex fixed combine schemes for regression include (weighted) median, geometric and generalized mean, decision templates etc.

Note that there are also techniques which combine the two approaches of model selection and model combination. The one thing that is common to these techniques is that they focus on selecting a subset of base models to be combined (by one of the above-mentioned schemes) in an ensemble, thus additionally improving the overall predictive performance. The improvement of hybrid approach is a consequence of the omission of poor performing base models, though it adds another level of complexity to the ensemble construction method and can be limited by the size of the available training data. A subset of such techniques include validate-and-select (Opitz & Shavlik, 1996), dynamic selection (Giacinto & Roli, 2000) and cascading classifiers (Gama & Brazdil, 2000).

2.2.2 Learning ensemble constituents

The second milestone in constructing ensembles is the algorithm(s) for learning the ensemble constituents. Based on how the constituents are learned, the ensembles can be categorized into two general categories: heterogeneous and homogeneous. In heterogeneous ensembles, the candidate base models are learned using different learning algorithms. In contrast, homogeneous ensembles is a category of ensembles where the base models are learned with the same learning algorithm, but from different training samples generated by manipulating the training data. Note, however, that on the intersection between heterogeneous and homogeneous ensemble methods lies one category which includes learning ensemble constituents by manipulating the parameters of the same learning algorithm (and manipulating the training data). In the remainder, we will give an overview of these categories of ensembles and describe some notable representative methods used in practice.

Procedurally, heterogeneous ensembles include base models learned by a variate of different learning algorithms (eg. decisions trees, SVMs, kNNs, artificial neural networks etc.), whose predictions are subsequently combined, resulting in an ensemble output. One of the popular methods in this category is stacking, which we covered in the previous section. Other notable methods for constructing heterogeneous ensembles include statistical ensembles (Tsoumakas, Katakis, & Vlahavas, 2004), catalog of models (Caruana, Niculescu-Mizil, Crew, & Ksikes, 2004) and cascade classifiers (Gama & Brazdil, 2000). These methods focus on selecting and combining base models, rather than learning the base models.

In contrast, homogeneous ensembles aggregate diverse and potentially unstable predictive models learned with the same learning algorithm (such as decision trees or artificial neural networks), that is, predictive models whose predictions vary sufficiently with small variations in the training data set. These variations in the training data set are generally obtained either by (1) manipulating the data instances, (2) manipulating the data features or (3) both.

Bagging (bootstrap aggregation) refers to one of the first and simplest methods, developed by Breiman (1996a), for constructing ensembles by manipulating the data instances. This method, depicted in Figure 2.5, employs bootstrap sampling with aggregation, where data instances are uniformly sampled with replacements to generate random samples (bootstrap replicates) of the training data, consequently used to learn a set of base models (ensemble constituents) from these replicates. The learned models are then combined by averaging their output (in the case of regression) or by voting (in the case of classification). Wagging is a variant of the bagging method, where the data instances are non-uniformly sampled, that is, a weight is stochastically assigned to each data instance which determines the probability of that instance to be included in the final bootstrap-replica.

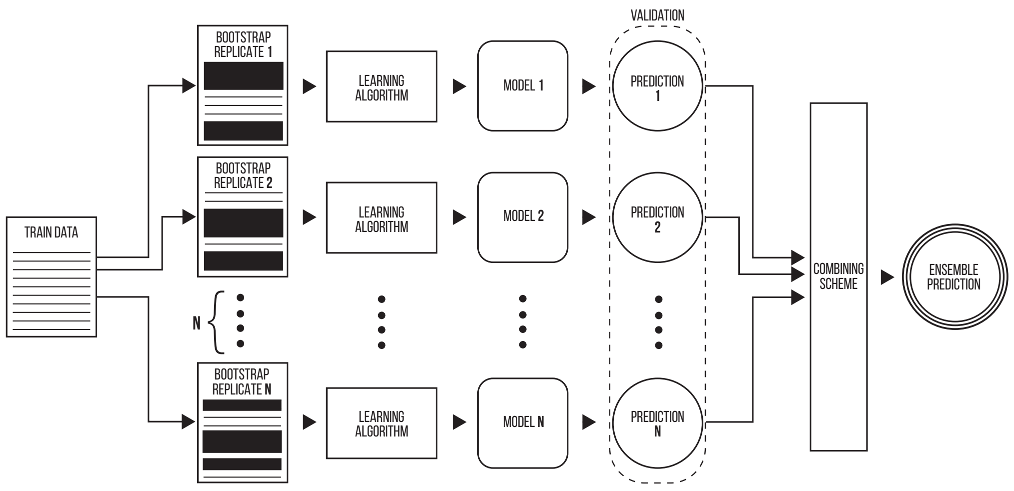


Figure 2.5: Illustration of the Bootstrap aggregation (Bagging) method for creating homogeneous ensembles by sampling data instances.

Another popular method which implements manipulation of the data instances is *boosting*. Boosting, introduced by Schapire (1990), refers to a general approach for obtaining an accurate prediction by combining several weak ones on a different distributions of the training data. The AdaBoost algorithm (Freund & Schapire, 1997), illustrated in Figure 2.6, is an implementation of the boosting approach for the task of classification. AdaBoost works iteratively; it uses different weighted distributions of the training data for learning the base models at each iteration. Depending on the outcome of past iteration this method decreases (for correct classification)/increases (for false classification) the weight value of every instance for the subsequent iteration of training the model. This process can assure that the weak predictors can focus on different instances, and thus creating more robust ones. Variants of AdaBoost include: BrownBoost (Freund, 1999), AdaBoost with confidence-rated predictions and arching. Moreover, LogiBoost (J. H. Friedman, Hastie, & Tibshirani, 1998) and the implementation of (Drucker, 1997) successfully tackles the problem of combining logistic and liner regressors, respectively, using the AdaBoost framework.

The methods that manipulate the data instances can often be ineffective when the training data contains few data instances or is relatively homogeneous. Moreover, constructing such ensembles can be very ineffective and computationally complex when the data dimensionality (the feature space) is high. The second type of homogeneous ensembles tackles these issues and incorporates predictive models learned from different subsets of the data features. The most notable method here is the *random subspace method* (RSM)

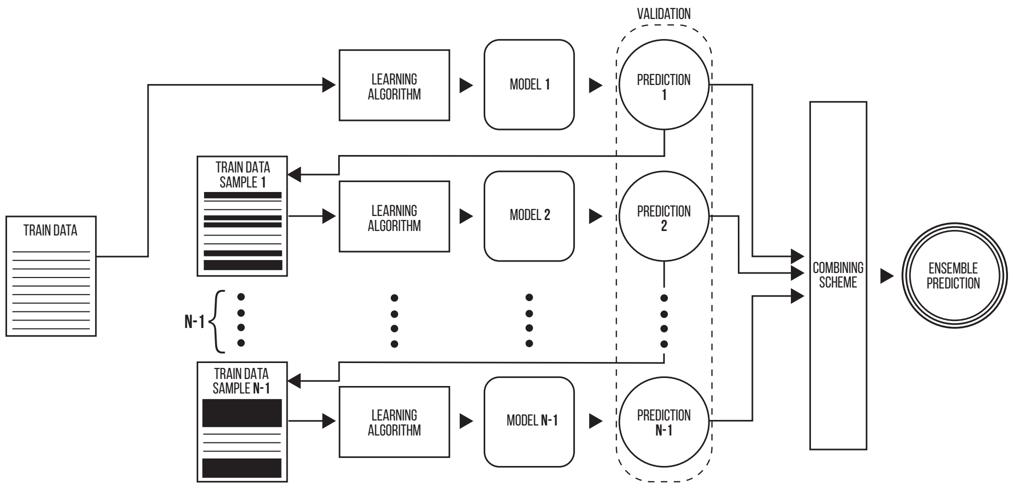


Figure 2.6: Illustration of the boosting (AdaBoost) method for creating homogeneous ensembles by sampling data instances.

(illustrated in Figure 2.7), developed by Ho (1998), which constructs ensembles of samples of the training data by sampling the feature space. Each ensemble constituent is learned on all data instances and a subspace of the original feature space. The predictions of the learned base models are then combined via standard combining schemes for classification or regression, i.e., voting schemes or averaging techniques, respectively. The RSM has been reported to perform well for problems where the data dimensionality is very high or when there is a certain redundancy in the feature space (Ho, 2000). The Rotation Forest method proposed by (Rodríguez, Kuncheva, & Alonso, 2006) also relies on feature sampling: It takes a subset of features at the beginning and applies PCA (Principle Component Analysis) on it, rotating the rest of the feature subsets before learning the next ensemble constituent. In contrast, instead of sampling features at random, more sophisticated methods are devolved for generating subsets of features based on evolutionary and genetic algorithms (Maclin & Opitz, 1999; Rokach, 2008).

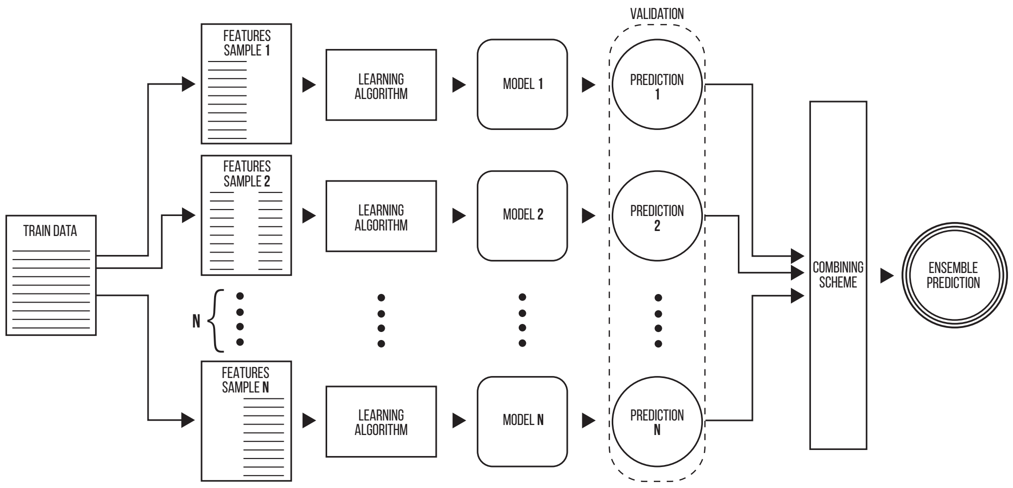


Figure 2.7: Illustration of the Random Subspaces method (RSM) for creating homogeneous ensembles by sampling data features.

Some methods exploit both dimensions of training data manipulation, and construct ensembles from samples of both data instances and data features. Panov and Džeroski (2007) propose the method Bagging of subspaces, where each ensemble constituent is learned simultaneously from a bootstrap replicate and a feature subset of the training data. This method, while being general in terms that it can handle any type of predictive model as a base model, it also leads to ensembles which are efficient to construct and have good predictive performance. The most popular method in this category (and probably overall), however, is *random forests* (RF)(Breiman, 2001).

Procedurally, the random forests ensembles consist of decision trees as base models, which are learned from different bootstrap replicas of the training data. However, in the process of growing the decision-tree constituents, this method at each node considers different (randomly drawn) subset of features from which the split is determined. The random forests ensembles have been proven to be both effective and computationally efficient, especially in cases when learned from high-dimensional training data (Fodor, 2002).

The final group of ensemble methods considers learning the base predictive models with the same algorithm, but with different parameters for each of them. The above-mentioned random forests method can also fall in this group, if a different random seed is used when constructing the base decision trees. Dietterich (2000a) proposes a similar method to the random forest called randomized C4.5. However, instead of choosing random subset of features (as in RF) for splitting the nodes in each tree, it first generates all possible splits at each node and randomly chooses one out of N top-ranked splits (in the original paper N is 20). Other notable methods in this group include Randomized FOIL (Ali & Pazzani, 1996) which employs rules as base models and ensemble of neural networks (Hansen & Salamon, 1990) where each base model is a neural network constructed with different parameters.

2.2.3 Interpretability of ensembles

For many real-world predictive tasks, a necessary requirement for a predictive model, in addition to its accuracy, is its interpretability. Understandable predictive models can provide a significant insight both into the learning process and into the domain at hand. In general, interpretability is an inherent deficiency of ensembles, given that they aggregate a set of models. However, several attempts are made that tackle this challenge which can be grouped in three general methodologies ,i.e. , (1) learning a meta-model , (2)selecting a model representative and (3) general explanation methodology.

The first methodology focuses on learning a complex meta-model while the ensemble is constructed as in the study of (Bridewell et al., 2005), where the authors propose a method which integrates the model structures of the ensemble constituents into a single model. Alternatively, meta-models can be learned a posteriori using the predictions of the ensemble constituents as a training data, as in stacking (Wolpert, 1992). The second methodology involves either selecting one of the base models to be a representative of the ensemble based on different criteria (accuracy, complexity or both) (Ferri-Ramírez, Flach, & Hernandez-Orallo, 2002); or learning a model representative from artificial data generated by the ensemble constituents (Craven, 1996; Domingos, 1998; Van Assche, 2008). The last methodology offers comprehensibility by computing and visualizing each features's contribution in the model's prediction (Štrumbelj & Kononenko, 2010). This contribution relates to the discrepancy (information difference, log-odds ratio or probabilities difference) between the initial model prediction and the average of different predictions with omitted subsets of features. This methodology has been successfully employed for interpreting "black-box" single models and ensembles applied to both classification tasks (Robnik-Šikonja & Kononenko, 2008) and regression tasks (Štrumbelj & Kononenko, 2011).

In contrast, Breiman (2001) further exploits the random forests method for feature ranking. However, while this approach provides additional insight into the problem at hand, it does not result in an interpretable ensemble. Moreover, the visualization tool RAFT (RAndom Forest Tool) (A. Cutler & Breiman, 2001), offers graphical aid for interpreting random forests analyses, albeit only for classification, by visualizing feature importance, outliers and different votes for each class using the proximity matrix obtained from the forest.

2.2.4 Ensemble performance

There is no generally accepted theory that unifies the concept of ensemble learning, which is not surprising given the variety of ensemble methods developed and their applications in a plethora of domains. Therefore this is still an open and active direction of research in the machine learning community. Strictly speaking, there are several theoretically sound explanations of the performance of the ensembles, however either they are assumption-bounded or focus on a narrow class of ensemble methods (Kuncheva, 2014; Ho, 2000; Hansen & Salamon, 1990; Hastie & Tibshirani, 1990). The general consensus, though, is that an ensemble should be constructed with accurate and diverse base models. In terms of accuracy, the base models should (at least) perform better than random guessing. In terms of diversity, the base models should be independent in making prediction, i.e., they should err differently when new (unseen) test data is presented to them. While intuitively these two requirements are sufficient for a well-performing ensemble, the findings of Kuncheva and Whitaker (2003) show that there is not always a strong correlation between them and the question of accuracy-diversity trade-off is more problem/task dependent than general. In contrast, Ali (1996) empirically shows that ensemble constituents should err dependently, however this diversity is negatively correlated to the overall ensemble performance. So then ... *Why do ensembles perform well?* And more importantly: *Why should ensembles be learned and applied to predictive tasks?*

There are two theories proposed which aim to answer the former question. The first theory (Vapnik, 1998; Schapire, 1990; Schapire et al., 1997; Allwein et al., 2000), formulates the task of ensemble learning in the framework of *large margin classifiers*. More specifically, it states that ensembles (especially boosting-alike ensembles) tend to expand the classification margins in the hypothesis space, thus enhancing their generalization capabilities. The second theory views the task of ensemble learning from a perspective of *bias-variance decomposition of error* (Geman et al., 1992; Kong & Dietterich, 1995; Breiman, 1996). Here, the performance of the ensembles is justified by their capability to reduce the variance or to reduce both bias and variance. However, Domingos (2000) joins these two theories by proving that they are equivalent: the notion of large margin classifiers can be expressed in the form of bias-variance and vice-versa. Following this insight, we will adopt the bias-variance decomposition theory as a framework for explaining the performance of the ensembles presented in this thesis.

Several studies (Dietterich, 2000b; Valentini, 2003; Džeroski et al., 2009) provide a few fundamental answers to the latter question. First, ensemble learning offers a very elegant solution to the model selection problem (Guyon, Saffari, Dror, & Cawley, 2010). While for learning a single predictive model the underlying learning algorithm solves an additional nontrivial task of model selection (the learned predictive model, besides having a good predictive performance, should also avoid under/over-fitting to the training data), ensemble learning methods tend to circumvent this task by combining a set of predictive models, thus minimizing the probability of a good performing predictive model being overlooked. The second reason for learning ensembles arises from the problem of limited coverage capabilities in the hypothesis space of learning algorithms. Namely, ensembles tend to expand the space of possible models learned, thus maximizing the chance for attaining optimal predictive base models. The result is improved generalization power of the ensemble, which provides a good approximation of the true solution.

Next, the learning algorithms, if viewed as search algorithms, often tend to end-up in a local optima which results in sub-optimal predictive models. Ensembles tend to lower this bias by boosting the algorithm towards the optimal solution and aggregating different sub-optimal predictive models. Last but not least, from practical point of view, learning ensembles tend to lower the computational complexity by decomposing the learning problem, i.e., ensembles can be learned from different samples of the training data, and even training data composed from subsets gathered from different places at different times.

Ensemble models have been successfully applied in many domains for tackling a variety of predictive tasks. A small subset of such applications includes applications in astronomy (Bazell & Aha, 2001), medicine (Steidl et al., 2010; Polikar et al., 2008), bioinformatics (Schietgat et al., 2010), ecology (D. R. Cutler et al., 2007; Kocev et al., 2013) etc. Ensembles have also been employed in the context of modeling dynamic systems, covered in the following section.

2.2.5 Ensembles of models of dynamic systems

Ensembles of models of dynamic systems have been previously considered in two different fashions. First, from machine learning perspective, the approach presented by Bridewell et al. (2005) considers ensembles of process-based models, where the structures of the ensemble constituents are integrated into a single meta-level model. The ensemble constituents are learned from different random training data samples. The single meta-level model is built in such a way that it includes the most frequent structure fragments (processes) in the base-level models. The results show that the resulting meta-level model still provides a process-based explanation of the observed system structure, while being more robust in terms of over-fitting. Note, however, that authors estimate the out-of-sample error of the models, by taking random sub-samples of the observed time-series data and removing them from the training data. Thus, the ability of the meta-level model to generalize outside the time span of the training data has not been considered nor evaluated.

In a similar context, Aleksovski et al. (2015) address the tasks of predictive modeling of discrete non-linear dynamic systems. Here, using external dynamic approach, the modeling problem is first transformed into a non-linear regression approximation problem subsequently tackled by learning fuzzy linear model trees and ensembles of fuzzy linear model trees. The results show that the ensembles improve the performance over the single fuzzy linear trees. Note that, this study focuses on short-term (one-step ahead) prediction of discrete-time dynamic systems, where the value of the time series in the next time-point is predicted, as opposed to long-term prediction of continuous non-linear dynamic systems.

Second, in a broader sense, ensemble predictions of dynamic systems are obtained by combining diverse predictions from an individual model, obtained either by perturbing the initial state of the model or varying (some of) the constant parameters. A proof of concept to such an approach is found in the practice of data assimilation (Kalnay, 2003). It is shown that using carefully chosen small amounts of information can recover the complete state of the system. Theory of synchronization in chaotic systems provides explanation to such approach, since linking chaotic systems with a single parameter can lead to synchronization of their states. A related recent development is the paradigm of supermodeling (Van den Berge et al., 2011; Mirchev et al., 2012).

A supermodel, or an ensemble of "imperfect" models, is comprised from a set of interconnected (coupled) models, which are integrated simultaneously and exchange information among themselves on a time-step basis. The idea of supermodeling comes from non-linear dynamics and the concept of attractor synchronization, where synchronized variations (in terms of constant parameters) of an individual model can substantially improve the model's stability. The supermodels, albeit fairly complex and consisting of typically small number of constituents (which are provided by a domain expert rather than learned), have been successfully employed in the context of climate modeling for addressing the task of long-term climate projections.

Part II

Methodology & Experimental Design

Chapter 3

Ensembles of Process-Based Models

In this chapter, we present the methodology for simulating and learning ensembles of process-based models, which is the main contribution of this thesis. We first present how ensembles of process-based models are simulated. Next, we present four different methods for learning a diverse set of ensemble constituents. Finally, we present the computational complexity of the proposed ensemble methods.

In order to simulate an ensemble, each base model needs to be simulated. The resulting ensemble output is a combination of the predictions of all individual base models. For obtaining a prediction for ensembles of process-based models, we use average, weighted average and weighted median as combining schemes, commonly used for regression tasks (Drucker, 1997). Section 3.1, presents an overview of the algorithm for obtaining predictions from ensembles of process-based models.

Next we describe the methodology for learning ensembles of process-based models, that is, learning diverse set of ensemble constituents. The underlying algorithm for learning process-based models employed in this thesis is ProBMoT, thus the proposed methodology aims at constructing homogeneous ensembles of process-based models. Recall from Section 2.2.2 from the previous chapter, homogeneous ensembles consist of base models that are learned with the same learning algorithm, but from different samples of the training data. The sampling approaches include: sampling of data instances, sampling of data features/attributes or both instances and features. The proposed methodology utilizes these sampling approaches, which results in four different methods for learning ensembles of process-based models. The first two methods of *bagging* and *boosting*, presented in Section 3.2 and Section 3.3, respectively, refer to learning ensembles from sampling data instance. More specifically, the proposed methods follow the key ideas of bagging (Breiman, 1996a) and Adaboost (Freund, 1999) in the context of time-series regression, and extend towards the process-based modeling paradigm.

The third method considers learning ensembles of process-based models by sampling data features. In this context, the task of generating random samples of the feature space from the traditional RSM (Ho, 1998) is implemented as a task of generating random subsamples of the library of domain knowledge. The algorithm of the *random library subsamples* method (RLS) is presented in Section 3.4. Finally, the last method that we propose deals with learning ensembles from both samples of data instances and data features. Section 3.5 presents an overview of the algorithm for *bagging random library subsamples* (BRLS) for learning diverse ensemble constituents from different samples of the data instances and different samples of the library of domain knowledge that is inspired by the work of Panov and Džeroski (2007).

3.1 Simulating Ensembles of Process-Based Models

In all cases, the real-valued predictions of the constituent models are combined per time-point, for each time-point separately. In the case of average, all base models participate in the resulting simulation equivalently. For weighted average and weighted median schemes, a confidence β is calculated for each of the base models based on their performance error. The base models with higher confidence will contribute more in the resulting ensemble simulation. The method for simulating an ensemble of PBMs is depicted in Algorithm 3.1.

Algorithm 3.1: Simulating ensembles of process-based models.

```
  Input: $ensemble, lib, D, scheme$
  Result: $\hat{y}_e$
1 $Simulations \leftarrow \varnothing$        /* simulations from ensemble constituents */
2 $\hat{y}_e \leftarrow \varnothing$          /* $\hat{y}_e$:  the resulting ensemble simulation */
3 forall $\{model, \beta\} \in ensemble$ do
4     $\hat{y} \leftarrow$ simulate($model, D$)
5     if inrange($\hat{y}, lib$) then
6         $Simulations \leftarrow Simulations \bigcup \{\hat{y}, \beta\}$
      end
7     else continue;
  end
8 switch $scheme$ do
      case $average$ do
          $\hat{y}_e \leftarrow$ average($Simulations$)
      end
      case $weighted\ average$ do
          $\hat{y}_e \leftarrow$ weightedAverage($Simulations$)
      end
      case $median$ do
          $\hat{y}_e \leftarrow$ median($Simulations$)
      end
      case $weighted\ median$ do
          $\hat{y}_e \leftarrow$ weightedMedian($Simulations$)
      end
  end
```

It takes as input: a set of process-based models denoted with *ensemble*, the library of domain knowledge *lib*, a data set D and a label *scheme* selecting the combination scheme used. The resulting prediction of the ensemble is a trajectory denoted with $\hat{y}_e$. First, each model from the set is simulated. The result of the prediction of an individual model for a data set D is a trajectory $\hat{y}$. Each model is accompanied with a confidence β, calculated based on the performance on a validation data set. We use this coefficient β in the weighted combining schemes. The pairs of trajectories and confidences $\{\hat{y}, \beta\}$ resulting from the simulation of the constituents in the ensemble is collected in the set *Simulations*.

In contrast to the task of obtaining an output from a regression model, where the resulting prediction is a single point for a given input, the task of predicting with process-based models is far more challenging. The simulation of a process-based model takes as input the initial values of the endogenous variables and the complete trajectories of the exogenous (forcing) variables. As output, it produces complete trajectories of the endogenous variables. In a predictive scenario, this can often lead to divergent trajectories and disastrous predictive misperformance. For this reason, we examine the simulated values of each prediction $\hat{y}$ whether they satisfy the range of constraints given in the library of background knowledge (line 5 in Algorithm 3.1). If a value from a prediction is outside the range specified in the library, the whole trajectory of that particular model is discarded, i.e., is not taken into account when calculating the resulting ensemble prediction. In this thesis, we use this kind of dynamic ensemble pruning as a standard technique when selecting the ensemble constituents and simulating the ensemble prediction. Finally, the valid simulations (denoted with *Simulations*) along with the respective confidence β_v are combined in the resulting ensemble prediction.

3.2 Bagging of Process-Based Models

The method for bagging process-based models is presented in Algorithm 3.2. The method takes four inputs: a library of domain knowledge *lib*, data consisting of training data D_T and validation data D_V, an incomplete model *incompleteModel*, and an integer k denoting how many base models are to be generated. The output is a set of process-based models denoted with *Ensemble*. Using `probmot`(line 4 in Algorithm 3.2), we learn a set candidate base models from different random samples D_S of the training data D_T. The `probmot` procedure follows the algorithm design principles of the process-based modeling paradigm, and resembles Algorithm 2.1 in terms of inputs, outputs and flow.

Algorithm 3.2: Constructing ensemble of process-based models with bagging.

```
   Input: $lib, \{D_T, D_V\}, incompleteModel, k$
   Output: $Ensemble$
1  $Ensemble \leftarrow \varnothing$                         /* set of base models */
2  for $i = 1$ to $k$ do
3      $D_S \leftarrow$ sampleData($D_T$)      /* randomly sample the training set $D_T$ */
4      $modelList_i \leftarrow$ probmot($lib, D_S, incompleteModel$)
5      $bestModel_i \leftarrow$ rank($modelList_i, D_V$)
6      $\beta_i \leftarrow$confidence($bestModel_i, D_V$)
7      $Ensemble \leftarrow Ensemble \bigcup \{bestModel_i, \beta_i\}$
   end
```

The notable difference from bagging in the context of regression is that in this case the data instances have a temporal ordering, which has to be retained in each data sample. To achieve this, we implement sampling by retaining the order of the instances by introducing a weight for each instance (time-point) that is provided as part of the data. The weight corresponds to the number of times the instance has been selected in the process of sampling with replacement (`sample` procedure). Instances that have not been selected (the ones with weight 0) are simply omitted from the sample.

To take into account the weights when learning a model from the sample, we implement the weighted root mean squared error (WRMSE) as an objective function in the process of parameter estimation:

$$WRMSE(m) = \sqrt{\frac{\sum_{t=0}^{N} \omega_t (y_t - \hat{y}_t)^2}{\sum_{t=0}^{n} \omega_t}},$$

where y_t and $\hat{y}_t$ correspond to the *measured* and *simulated* values (simulating the base model m) of the system variable y at time point t, N denotes the number of instances in the data sample, and ω_t denote the weight of the data instance at time point t.

Algorithm 3.3: Procedure for calculating the confidence of individual ensemble constituents.

```
   Input: $model, D$
   Output: $\beta$
 1 let $\hat{y}$                                    /* simulated system variable $y$ */
 2 let $y$                                           /* measured system variable $y$ */
 3 $\hat{y} \leftarrow$simulate($model, D$)
 4 $MD \leftarrow \sup(|y_t - \hat{y}_t|)^2, t = 0..N$   /* calculate maximum          */
                                                  discrepancy between
                                                  measurements $y$ and
                                                  simulation $\hat{y}$, where $N$ is
                                                  number of time-points in $D$
 5 $L_t \leftarrow \frac{|y_t - \hat{y}_t|^2}{MD}, t = 0..N$   /* calculate square loss at time point $t$ */
 6 $\bar{L} \leftarrow \frac{1}{N}\sum_{t=0}^{N} L_t$                   /* calculate average loss */
 7 $\beta \leftarrow \frac{\bar{L}}{1 - \bar{L}}$                     /* calculate average confidence */
```

The output of a modeling task, when using ProBMoT, is a list of process-based models, which is a posteriori sorted according to their performance (line 5 in Algorithm 3.2). Depending on the input in the method, this ranking can be based either on the performance on a separate validation data set D_V, or on the training sample (if $D_V == D_T$). The highest ranked model from each modeling task i (out of k) denoted as $bestModel_i$, is selected to be an ensemble constituent in the output *Ensemble*.

Note that each ensemble constituent is paired with its own confidence β. The procedure for calculating the confidence, presented in Algorithm 3.3, takes 2 inputs: the highest ranked model returned by ProBMoT and a data set D. Fist the *model* is simulated on the data set D resulting in a trajectory $\hat{y}$. Based on the error at each time point in the trajectory an average loss $\bar{L}$ is calculated for the *model* (6 in Algorithm 3.2). From this loss, a confidence measure β is derived, where low values of β denote high confidence. The β coefficient is an indicator of the performance of the base model and is used in the process of simulating the ensemble, i.e., combining the simulations of the constituents into an overall ensemble prediction.

3.3 Boosting of Process-Based Models

The method for boosting of process-based models is presented in Algorithm 3.4. In analogy to the previous method for bagging process-based models, it takes the same four inputs: a library of domain knowledge *lib*, data consisting of training data D_T and validation data D_V, incomplete model *incompleteModel*, and an integer k denoting how many base models are to be generated. In contrast to bagging, however, here we start with the complete training data set instead of a random bootstrap sample. To account for the (re)sampling of the dataset for each succeeding boosting iteration, we use the same concept of weighting each data/time-point. However here, instead of uniformly random choosing the weights as in bagging, the weights are (re)calculated after every boosting iteration (line 8 in Algorithm 3.4). The value of the weights directly relates to the error made by the best performing model (measured on the training data) from the preceding iteration at the respective time point. In the learning phase, we also use the WRMSE objective function, presented in Eq. 7.

Algorithm 3.4: Constructing ensemble of process-based models with boosting.

```
   Input: $lib, \{D_T, D_V\}, incompleteModel, k$
   Output: $Ensemble$
 1 $Ensemble \leftarrow \varnothing$                         /* set of base models */
 2 $\omega_t \leftarrow 1, t = 0..N$                    /* $\omega_t$ is the weight of time */
                                                point $t$, where $t = 0..N$ and
                                                $N$ is the number of
                                                measurements in $D_T$
 3 for $i = 1$ to $k$ do
 4     $modelList_i \leftarrow$ probmot($lib, D_S, incompleteModel, \omega$)
 5     $bestModel_i \leftarrow$ rank($modelList_i, D_V$)
 6     $\beta_i \leftarrow$confidence($bestModel_i, D_V$)
 7     $Ensemble \leftarrow Ensemble \bigcup \{bestModel_i, \beta_i\}$
 8     $\omega \leftarrow$reweight($model_i, D_T, \omega$)
   end
```

Algorithm 3.5: Procedure for calculating the training sample of the boosting iteration.

```
   Input: $model, D, \omega$
   Output: $\omega$
 1 let $\hat{y}$                                /* simulated system variable $y$ */
 2 let $y$                                      /* measured system variable $y$ */
 3 $\hat{y} \leftarrow$simulate($model, D$)
 4 $MD \leftarrow \sup(|y_t - \hat{y}_t|)^2, t = 0..N$     /* calculate maximum */
                                                discrepancy between
                                                measurements $y$ and
                                                simulation $\hat{y}$, where $N$ is
                                                number of time-points in $D$
 5 $L_t \leftarrow \frac{|y_t - \hat{y}_t|^2}{MD}, t = 0..N$    /* calculate square loss at time point $t$ */
 6 $\bar{L} \leftarrow \sum_{t=0}^{N} L_t \frac{\omega_t}{\sum_{t=0}^{N} \omega_t}$      /* calculate weighted average loss */
 7 $\beta \leftarrow \frac{\bar{L}}{1 - \bar{L}}$                 /* calculate average confidence */
 8 $\omega_t \leftarrow \omega_t \beta^{1-L_t}, t = 0..N$                  /* update weights */
 9 $\omega \leftarrow$normalize($\omega, N$)                /* normalize weights to $N$ */
```

More specifically, the weighting procedure, presented in Algorithm 3.5, takes 3 inputs: the highest ranked model (denoted with *model*) from the previous iteration, a data set D, and the respective set of weights ω. While this function resembles the `confidence` function, there are important differences: Here we consider a set of time-point wise weights and loss (rather than a single overall loss), and we calculate this on the training data (in contrast to validation data). First, the *model* is simulated on the data set D. Next, based on the error at each time point in the trajectory and the set of weights ω, the weighted average loss $\bar{L}$ is calculated. Finally, the set of weights is updated: the smaller the loss, the more the weight is reduced – focusing on harder parts of the data set in the future iterations of the algorithm.

The output of the *boosting* method is also a set of pairs (process-based models and their respective confidences) denoted with *Ensemble*. The highest ranked process-based model from each boosting iteration is considered as an ensemble constituent, for which a confidence is calculated. As in bagging, the ranking can be based on the performance of the process-based model on a separate validation data set D_V, or on the training sample (if $D_V == D_T$).

3.4 Learning Process-Based Model Ensembles via Random Library Subsamples

The method for learning ensembles of process-based models from random library subsamples (RLS) is presented in Algorithm 3.6. The method takes five inputs: a library of domain knowledge (*lib*), a dataset consisting of training data (D_T) and validation data (D_V), an incomplete model (*incompleteModel*), a boolean variable (*allowDuplicates*), and an integer k denoting how many base models are to be generated. The output is a set of process-based models denoted with *Ensemble*.

For the task of sampling the library (line 3 in Algorithm 3.6), the process alternatives are randomly sampled (excluded) from the original library. The sampling algorithm takes as input the complete library and enlists all the process templates defining more than one modeling choice. In turn, for each process template on the list, it takes a random sample of the available modeling choices to be included in the sampled library. Note that the library sampling does not assume a uniform distribution of samples: the probability of a library sample is proportional to the size of the induced space of candidate models. In particular, the probability of a library sample lib_S of the whole library lib equals

$$P(lib_S) = \frac{|L_S|}{\sum\limits_{L_i \in \mathbb{P}(L)} |L_i|},$$

where L and $L_S \subseteq L$ correspond to the sets of candidate models induced by lib and lib_S (for a given incomplete model specification), respectively. Moreover, $|\cdot|$ denotes set cardinality and $\mathbb{P}(L)$ denotes the powerset of L, i.e., the set of all the possible subsets of L. For example, suppose we learn an ensemble by sampling the library from Table 2.1 from Section 2.1.4. This library has two template processes with process-alternatives, i.e., the template processes GrowthRate and Respiration. Therefore, this library can be sampled into nine different valid sub-libraries: one (the original) from which eight candidate models are induced, two libraries (by omitting one of the two alternatives of Respiration) which result in four candidate models each, two (by omitting one of the two alternatives of GrowthRate) which result in two candidate models each, and the remaining four resulting in one candidate model each. The last four sub-libraries which employ one process alternative (out of the two possible) are less likely to to be selected/generated (1/24 each) than the rest sub-libraries where the probability is 1/3 for the first one, 1/6 for the next two, and 1/12 for the last remaining two.

Given that the method always takes as input the same original library, there is a high probability of learning and choosing identical models from different library samples, thus filling the ensemble constituent set with multiple copies of the same model. To account for this, the method incorporates two different alternatives for generating the ensemble constituent set, i.e., with and without duplicates. For the former, duplicates are allowed

Algorithm 3.6: Constructing ensemble of process-based models with random library subsamples

```
   Input: $lib, \{D_T, D_V\}, incompleteModel, allowDuplicates, k$
   Output: $Ensemble$
1  $Ensemble \leftarrow \varnothing$                         /* set of base models */
2  repeat
3      $lib_S \leftarrow$ sampleLib($lib$)          /* randomly sample the library $lib$ */
4      $modelList_i \leftarrow$ probmot($lib_S, D_T, incompleteModel$)
5      $bestModel_i \leftarrow$ rank($modelList_i, D_V$)
6      $\beta_i \leftarrow$confidence($bestModel_i, D_V$)
7      if $allowDuplicates$ then
8          $Ensemble \leftarrow Ensemble \bigcup \{bestModel_i, \beta_i\}$
       end
9      else if $bestModel_i \notin Ensemble$ then
10         $Ensemble \leftarrow Ensemble \bigcup \{bestModel_i, \beta_i\}$
       end
   until size($Ensemble$)$\neq k$;
```

in the constituent set (line 7 in Algorithm 3.6). k library samples are generated (with k denoting number of ensemble iterations), and the best model out of each modeling task is chosen to be an ensemble constituent, regardless of whether that particular model was already in the constituent set or not. For the latter, to incorporate more diversity in the ensemble, the method generates library samples (and performs modeling tasks) until the resulting ensemble contains k distinct constituents. Here, at line 9 in Algorithm 3.6 the method first checks whether the best learned model has already been included in the ensemble.

Analogously to the previous methods, the output of the *RLS* method is a set of pairs model-confidence denoted with *Ensemble*. The highest ranked process-based model learned from each subsample of the library is considered as an ensemble constituent, for which a confidence is calculated. Similarly, the ranking evaluation can be performed on a separate validation data set D_V, or on the whole training data set (if $D_V == D_T$).

3.5 Learning Process-Based Model Ensembles via Bagging of Random Library Subsamples

The last method that we propose combines the methods of bagging and random library subsamples. The algorithm for learning ensembles of process-based models with the bagging random library subsamples (BRLS) method is presented in Algorithm 3.7. The method takes four inputs: a library of domain knowledge (*lib*), a dataset consisting of training data (D_T) and validation data (D_V), an incomplete model (*incompleteModel*) and an integer k denoting how many base models are to be generated.

At each iteration, the BRLS algorithm learns a process-based model from different samples of the training data and the library. For the task of randomly sampling the data instances, we use the same procedure as in bagging. Analogously, generating different subsamples of the library of domain knowledge, we reuse the probability sampling procedure developed for the RLS method. BRLS is in the same framework with the previous

Algorithm 3.7: Constructing ensemble of process-based models with bagging of random library subsamples

```
  Input: $lib, \{D_T, D_V\}, incompleteModel, k$
  Output: $Ensemble$
1 $Ensemble \leftarrow \varnothing$                              /* set of base models */
2 for $i = 1$ to $k$ do
3     $lib_S \leftarrow$ sampleLib($lib$)             /* randomly sample the library $lib$ */
4     $D_S \leftarrow$ sampleData($D_T$)        /* randomly sample the training set $D_T$ */
5     $modelList_i \leftarrow$ probmot($lib_S, D_S, incompleteModel$)
6     $bestModel_i \leftarrow$ rank($modelList_i, D_V$)
7     $\beta_i \leftarrow$confidence($bestModel_i, D_V$)
8     $Ensemble \leftarrow Ensemble \bigcup \{bestModel_i, \beta_i\}$
  end
```

three algorithms, thus the output, denoted with *Ensemble*, consists of pairs of process-based models and their respected confidences that are either calculated using a separate validation data set D_V, or on the training sample (if $D_V == D_T$).

3.6 Complexity of Constructing Ensembles of Process-Based Models

In order to properly assess the computational complexity of learning ensembles of process-based model, we first need to establish the complexity of learning a single model that will serve as a base line for comparison. Recall from Section 2.1.4 that the algorithm for learning process-based models consists of two main sub-tasks: enumerating all possible model structures, and estimating the parameters of each of them.

Fig 3.1 A presents a diagram of relative execution times for each of the tasks through the prism of learning an example population dynamics model as described in Section 2.1.4. The first task, the structure enumeration process (red box), is a traversal algorithm through the space of model components, which is linear to the resulting number candidate models N (Čerepnalkoski, 2013). In this example, it results in 8 candidate models. Second, for each of these candidates, a parameter estimation task is performed (blue box), the efficiency of which is related to the number of parameters each candidate model has and the number of observed time points. Note that, even though the overall complexity of the parameter estimation task is $\mathcal{O}(N)$, in most practical cases more than 99% of the computational time is spent in this phase. At the end of each modeling task, we additionally rank (yellow box) the learned models based on their performance on training/validation dataset. The ranking task is performed by an insertion sort algorithm based on the models' performances and has a complexity of $\mathcal{O}(N logN)$ (given that sorted list is maintained after every model generation), where N is the number of candidate models. In this thesis we define the complexity of one iteration of ProBMoT, i.e. obtaining one process-based model as the benchmark unit for assessing the complexity of learning different ensembles of process-based models.

Now, for analyzing the computational complexity of the different methods for learning ensembles of process-based models, suppose we learn ensembles with five constituents for the same modeling task of population dynamics. Fig 3.1 B presents the computational time needed for learning an ensemble using the bagging method. It is essentially a repetition of the tasks needed for learning a single model for every ensemble constituent (separated by dashed line), with the exception of two additional tasks: one for sampling the training data (gray box) at the beginning of each ensemble iteration, and one for simulating the ensemble at the end (green box).

Even though the figure depicts the serial implementation of bagging, this algorithm handles the processes of learning different base models completely independently. Therefore, it can be parallelized to handle different tasks with different bootstrap replicates on different CPUs, which is very useful, performance-wise, for computationally intensive learning tasks such as process-based modeling. The complexity of learning an ensemble with boosting (Fig 3.1 C) resembles the one of bagging, though, the boosting algorithm cannot be parallelized, as each new boosting iteration depends on the outcome of the previous one. This makes such ensembles inefficient to learn, which is strongly felt when large libraries are considered or/and the tasks involve learning many constituents.

The next method that we investigate is learning ensembles with the random library subsamples method (Fig 3.1 D). While the algorithm (as presented in Algorithm 3.6) is iterative, we can implement it much more efficiently. Instead of sampling the space of components (i.e., sampling the domain knowledge) and running ProBMoT with each sample of the library, we can sample the generated search space and choose from the candidate models. First we generate all the models from the original library and fit their parameters. Next, we generate all the necessary library samples (orange box), and perform the task of searching and sorting models which are determined by the particular subsample of the library.

By transforming the sampling problem from sampling domain knowledge to sampling the model structures from the complete search space, we minimize the number of ProBMoT runs (to one), consequently substantially gaining computational efficiency, as compared to the other three ensemble methods using ProBMoT. Recall that selecting the constituent set for such ensembles can be performed in two ways: with and without duplicates. For the former, the execution time is correlated to the number of iterations needed (for this example is five). For the latter, the execution time depends on the random generator: It will either learn the constituents (in the best case) from as much libraries as iteration or (in the worst case) it will finally populate the constituent set with the last possible library. For this example, it can take from a minimum of five (yellow boxes) to a maximum of nine iterations (additional four opaque-yellow boxes). Finally, at the end an ensemble simulation task is performed.

The last method proposed in this thesis is learning ensembles by bagging of random library subsamples (Fig 3.1 E). This method in addition to learning the base models from different library subsamples, it also learns them from different samples of the training data. Here, first all the possible model structures are enumerated, followed by the procedure for constructing different library subsamples. Each of these subsamples is then given as an input to the PBM algorithm, which also takes as input a sample of the training data. Even though the BRLS cannot be as efficiently implemented as the RLS method, it can be parallelized, that is, every ensemble constituents can be learned on different CPU. This ability, coupled with the (smaller) subsamples of the library, which yields smaller sets of candidate models, gives the BRLS method a significant computational advantage over both the methods of bagging and boosting.

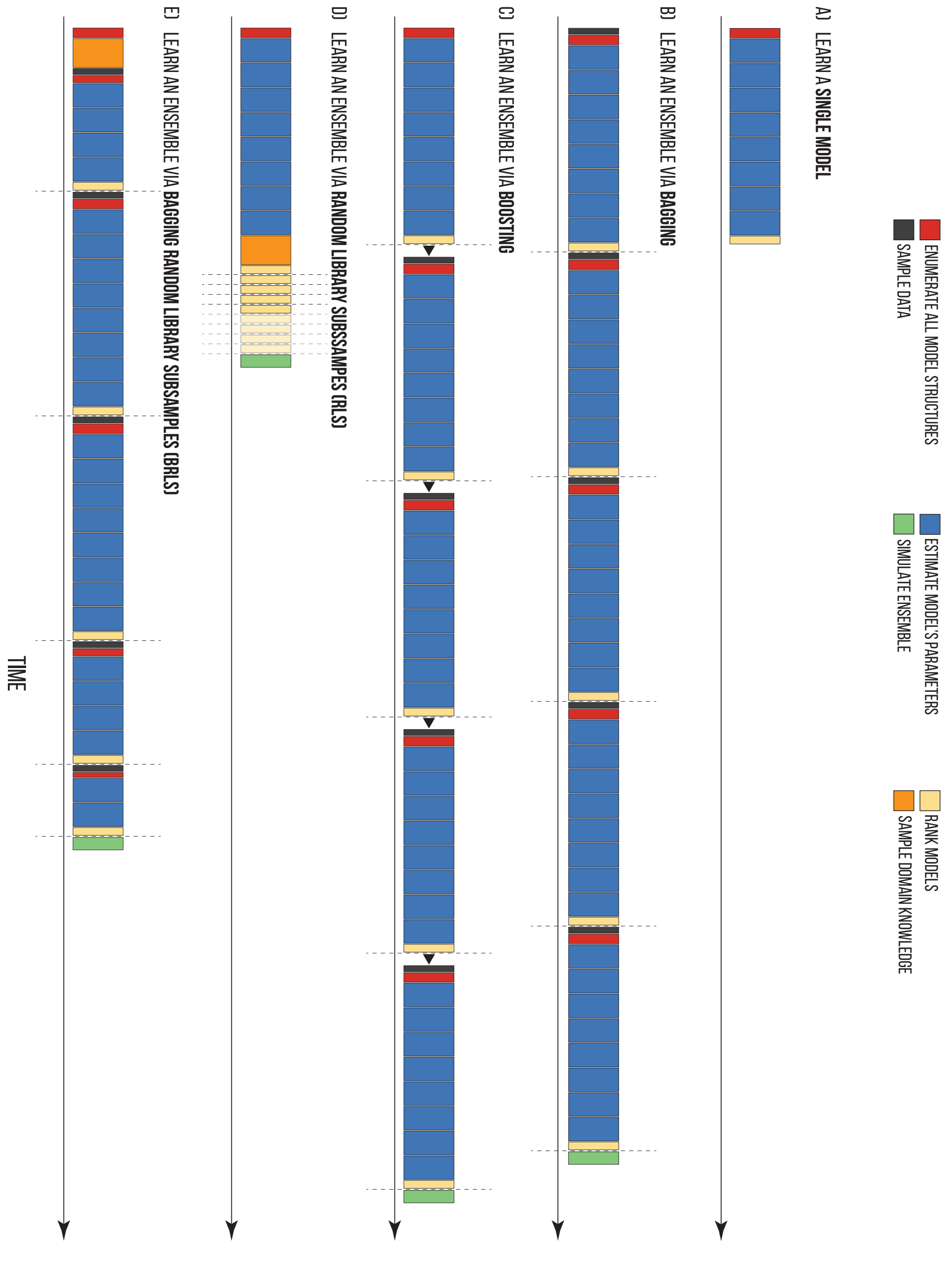

ENUMERATE ALL MODEL STRUCTURES
SAMPLE DATA
ESTIMATE MODEL'S PARAMETERS
SIMULATE ENSEMBLE
RANK MODELS
SAMPLE DOMAIN KNOWLEDGE
A) LEARN A SINGLE MODEL
B) LEARN AN ENSEMBLE VIA BAGGING
C) LEARN AN ENSEMBLE VIA BOOSTING
D) LEARN AN ENSEMBLE VIA RANDOM LIBRARY SUBSSAMPES (RLS)
E) LEARN AN ENSEMBLE VIA BAGGING RANDOM LIBRARY SUBSAMPLES (BRLS)
TIME

Figure 3.1: A graphical representation of the time needed to learn a single process-based model and ensembles of process-based models with five constituents by bagging, boosting, library sampling, and bagging of library samples using the library presented in Table 2.1 from Section 2.1.4.

Chapter 4

Experimental Design

In this chapter, we present the setup of experiments used to evaluate the predictive performance of the ensembles of process-based models. In the following sections, we first introduce the data sets to be used in the experiments, then briefly describe the two other ProBMoT inputs, i.e., the library of modeling knowledge and incomplete models, and finally define the performance metrics used to assess the process-based models and ensembles thereof.

4.1 The Data

The real-world data used in this study originates from three aquatic ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan and Lake Zurich in Switzerland.

Lake Bled is located in the Julian Alps in north-western Slovenia and occupies an area of 1.4 km^2, with a volume of 0.0257 km^3, a maximum depth of 30.1 m and an average depth of 17.9 m. The measurements, performed by the Slovenian Environment Agency, consist of physical, chemical and biological data for the period from 1996 to 2002. All the measurements were performed once a month and depth-averaged for the upper 10 m of the lake. To obtain daily approximations, the data was interpolated with a cubic spline algorithm and daily samples were taken from the interpolation (Atanasova, Todorovski, Džeroski, Remec, et al., 2006).

Lake Kasumigaura is located 60 km to the north-east of Tokyo, Japan ($36.0403°N$, $140.3942°E$). It has an average depth of 4 m, a volume of 662 million cubic meters, and a surface area of 220 km^2. The data set comprises monthly measurements in the period from 1986 to 1992. Again, to obtain daily approximations, the measurements were interpolated using linear interpolation and daily samples were taken from the interpolation (Atanasova, Recknagel, et al., 2006).

Lake Zurich is located in the south-western part of the canton of Zurich in Switzerland ($42.1970°N, 88.0934°W$). It has an average depth of 49 m, volume of 3.9 km^3 and a surface area of 88.66 km^2. The data comprises measurements performed by the Water Supply Authority of Zurich in the period from 1996 to 2002. The measurements, taken once a month, include profiles of physical, chemical and biological variables from 19 different sites. They were weight averaged to the respective epilimnion (upper ten meters) and hypilimnion (bottom ten meters) depths. The data was interpolated with a cubic spline algorithm and daily samples were taken from the interpolation (Dietzel, Mieleitner, Kardaetz, & Reichert, 2013).

We use the same structure of population dynamics model in all three aquatic ecosystems. It includes a single equation (ODE) for a system variable representing the phytoplankton biomass (measured as *chlorophyll-a* in Lake Kasumigaura). The exogenous variables include the concentration of zooplankton *Daphnia hyalina* (available only for Bled and Zurich), dissolved inorganic nutrients of nitrogen, phosphorus, and silica (ammonia in Lake Kasumigaura), as well as two input variables representing the environmental influence of water temperature and global solar radiation (light).

Table 4.1: Overview of the data used for modeling population dynamics in the three lakes: Lake Bled, Lake Kasumigaura and Lake Zurich.

	Bled	Kasumigaura	Zurich
Environmental influence	Temperature Light	Temperature Light	Temperature Light
Nutrients	Phosphorus Nitrogen Silica	Phosphorus Nitrogen Ammonia	Phosphorus Nitrogen Silica
Primary producer	Phytoplankton	Phytoplankton	Phytoplankton
Zooplankton	*D. hyalina*	None	*D. hyalina*
Training data (labels)	1996-2000 (B1–B5)	1986-1990 (K1–K5)	1996-2000 (Z1–Z5)
Validation data	2001	1991	2001
Test data	2002	1992	2002

Table 4.1 provides a summary of the data sets we used in the experiments. For each aquatic ecosystem, we used seven data sets corresponding to the last seven years of available measurements. Five of these were used (one at a time) for training the base-models, one was used for validating the models in the process of selecting the ensemble constituents, and one was used to measure the predictive performance of the learned models and ensembles.

In each learning experiment, we take a single (year) training data set, learn a single model or an ensemble using the training and the validation data set, and test the predictive performance of the learned models on the test data set. We therefore perform 15 experiments. In the tables, we label them by the label of the training set used, which is comprised of the initial letter of the lake name, followed by a digit for 1 to 5 corresponding to each of the 5 consecutive years of measurements. The labels are therefore B1–B5, K1–K5 and Z1–Z5, and B5, for example, denotes the measurements for Lake Bled taken in the fifth year, i.e., the year 2000.

4.2 Modeling Knowledge

In the experiments performed, we use the library of domain knowledge for modeling population dynamics in aquatic ecosystems, presented by Čerepnalkoski et al. (2012), Čerepnalkoski (2013). This library is based on the previous work of Atanasova, Todorovski, Džeroski, and Kompare (2006). The library of domain knowledge, combined with the modeling assumptions, results in 18144 candidate models for Lake Kasumigaura and 27216 candidates for the other two lakes.

Figure 4.1 illustrates a high-level overview of the library for modeling population dynamics in aquatic ecosystems. The library organizes the templates in hierarchies. The hierarchy of entity templates (represented by thick horizontal lines in Figure 4.1) in aquatic ecosystems includes the `Ecosystem` entity and the `Environment` entity templates at the highest level. The `Ecosystem` entity template instantiates further on into `Population`

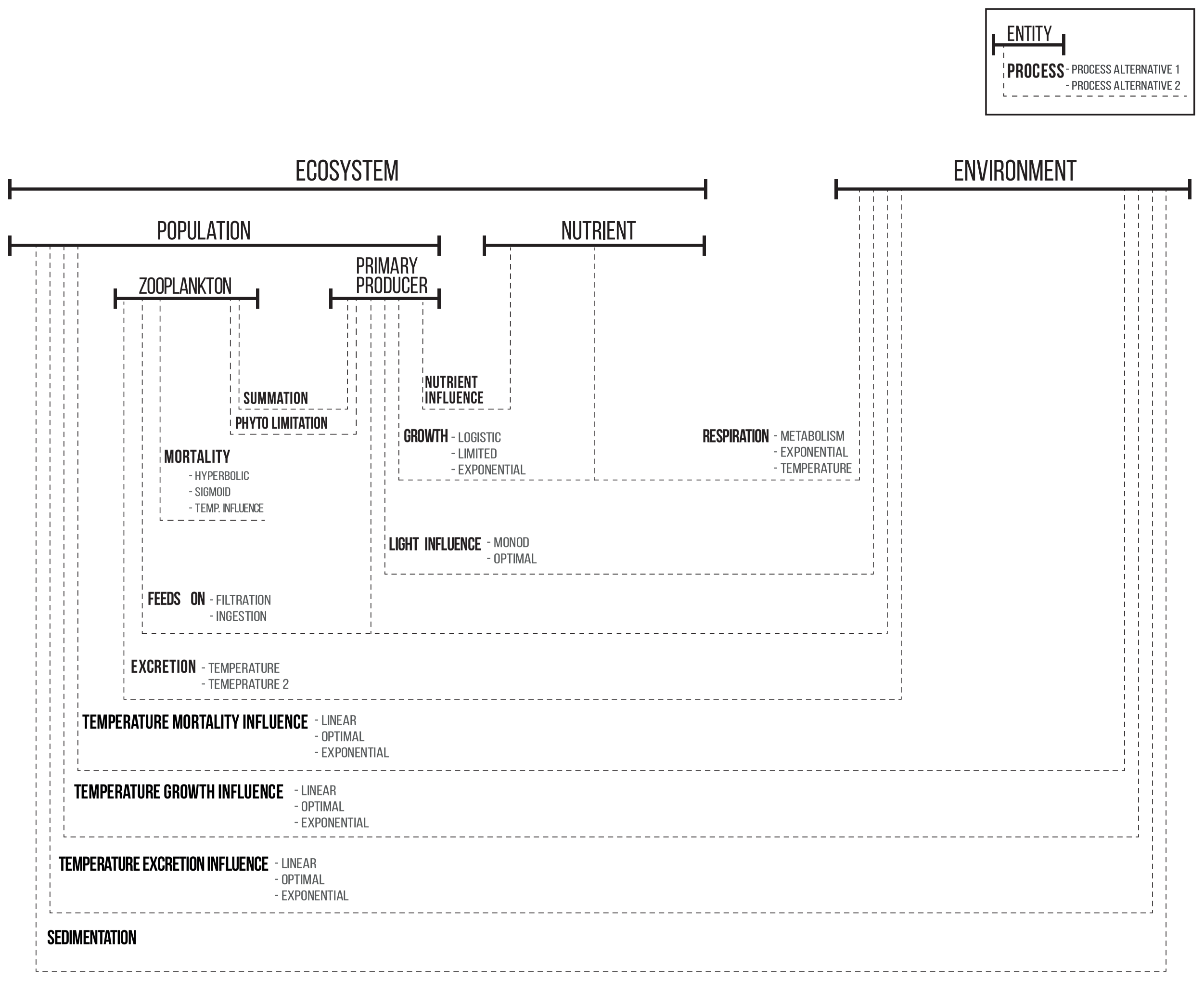


Figure 4.1: The library of modeling knowledge comprising template entities (thick horizontal lines) and processes (dashed lines connecting the entities) for modeling population dynamics in aquatic ecosystems.

and ***Nutrient*** entity templates, the first being further specialized into ***Zooplankton*** and ***Primary producer*** entity templates.

Similarly, the process templates (represented by dashed lines, connecting the entity templates) are also organized into a hierarchy. This hierarchy defines the space of modeling alternatives. For example, the process GROWTH which defines the interaction of growth of the ***PrimaryProducer*** influenced by the ***Nutrient*** and the ***Environment***, is high in the hierarchy of process templates. This process template can be further instantiated to one of the possible alternatives of LOGISTIC, EXPONENTIAL or LIMITED which define the rate of growth. It is similar with the process template for the "inverse" interaction of ***Respiration***, which can be influenced by METABOLISM or TEMPERATURE or can even have an EXPONENTIAL trend. The complete library of domain knowledge for modeling population dynamics in aquatic ecosystems, presented in the process-based modeling formalism, is given in Appendix A.

Note, however, an important difference between the setup of the bagging and boosting experiments and the experiments performed with the methods of random library subssamples and bagging of random library subsamples. In the last two, we use the whole library of domain knowledge as described previously. The use of such library is prohibitive for bagging and boosting, due to the high computational complexity considering the large space of candidate models in each learning iteration. To address this issue, we use a simplified version of the original library that results in 320 candidate model structures for the two Lake Kasumigaura and Artificial Lake Kasumugaura; and 128 candidates for the rest of the lakes. We carefully prepared the simplified library, omitting only modeling alternatives (process templates) that are rarely observed to be among the top-ranked models in the single-model experiments with ProBMoT. The issue of computational complexity of learning ensembles of PBMs was already discussed in Section 3.6 in the previous chapter.

4.3 ProBMoT Settings

ProBMoT implements the Differential Evolution (DE) (Storn & Price, 1997) method for parameter estimation. For the experiments performed in this paper, the DE parameters were set as follows: a population size of 50, strategy *rand/1/bin*, differential weight (F) and the crossover probability (Cr) of 0.6. The limit on the number of evaluations of the objective function is one thousand per parameter. For simulating the ODEs, we used the CVODE simulator (Cohen & Hindmarsh, 1996) with absolute and relative tolerances set to 10^{-3}.

4.4 Performance Evaluation Metrics

To evaluate the predictive performance of a given model m, we use the measure of relative root mean squared error (*ReRMSE*) (Breiman et al., 1984), defined as:

$$ReRMSE(m) = \sqrt{\frac{\sum_{t=0}^{n}(y_t - \hat{y_t})^2}{\sum_{t=0}^{n}(\bar{y} - \hat{y_t})^2}},$$

where n denotes the number of measurements in the test data set, y_t and $\hat{y_t}$ correspond to the *measured* and *predicted*[1] value of the system variable y at time point t, and $\bar{y}$ denotes the mean value of y in the test data set. Note that the usual root mean squared error is observed here relative to the standard deviation of the system variable in the test

[1]Predictions are obtained by simulating the model m on the test set.

data, hence allowing us to compare the errors of models for different system variables with measured values on different scales (e.g., phytoplankton in different lakes).

4.4.1 Evaluating diversity

To evaluate the conjecture that the power of ensembles is based on the exploitation of the diversity of the ensemble constituents, we measure the diversity of the ensemble constituents and correlate it to the performance improvement of ensembles over single models. To measure the diversity of the base-models in the ensemble e, we measure the average pairwise difference of the base model simulations

$$\mathit{Diversity}(e) = \frac{1}{\binom{|e|}{2}} \sum_{\{m_1,m_2\}\subset e} \sqrt{\frac{\sum_{t=0}^{n}(y_{1,t} - y_{2,t})^2}{n}},$$

where $|e|$ denotes the number of base-models in the ensemble e, n the number of measurements in the data set, m_1 and m_2 two models from e, and $y_{1,t}$ and $y_{2,t}$ the simulated values of these models at time point t. To assess the performance improvement of the ensemble e over a single model m, we calculate

$$\mathit{Improvement}(e,m) = -\frac{\mathit{ReRMSE}(e) - \mathit{ReRMSE}(m)}{\mathit{ReRMSE}(m)},$$

where the model m is learned from the complete training set, and the base-models from the ensemble e are learned on different bootstrap samples of the training set. We draw a scatter plot that depicts the correlation between ensemble diversity and performance improvement. Also we calculate both the Pearson Correlation Coefficient and Spearman's Rank Correlation Coefficient between them, where the former indicates whether there is a linear relation and the latter whether the relation between the performance improvement and the intra-ensemble diversity has a monotonic trend.

4.4.2 Statistical comparison of performance

We observe and compare the predictive performance (in terms of $ReRMSE$) of the models learned using different algorithms on the 15 experiments. To properly assess the significance of the differences between the performances of models obtained with different algorithms, we follow the standard statistical procedure recommended by Demšar (2006). We use the corrected (Iman & Davenport, 1980) Friedman test (M. Friedman, 1940), followed by two post-hoc tests: the Nemenyi test (Nemenyi, 1963) and the Bonferroni-Dunn test (Dunn, 1961). A positive outcome of the Friedman test indicates difference between the performances of the different algorithms considered. After the completion of the Friedman test, we proceed with performing post-hoc tests to identify which differences are statistically significant.

The first post-hoc test, i.e, the Nemenyi post-hoc test, computes the critical distance between the algorithm ranks at a given level of statistical significance (in our case, we set the significance level threshold at $95\%, p = 0.05$). Only differences of the average ranks larger than the critical distance are considered significant; for those, we can claim that one algorithm outperforms (i.e., performs significantly better than) the other. This test is performed to obtain an assessment of the relative performance of the methods considered. In this paper, we perform this test to compare different design decisions for the newly proposed method. The results of the Friedman-Nemenyi tests are depicted by average rank diagrams, where the critical distance is shown as a solid red line.

The second post-hoc test, i.e., the Bonferroni-Dunn post-hoc test, is performed to test how a proposed method performs in a comparison to other methods. This test is similar to the Nemenyi test, where a critical distance between the algorithm ranks is computed at a given level of significance (in this paper the significance level threshold is at $95\%, p = 0.05$), which denotes how one method (i.e, an ensemble learned using the library sampling method) compares to the other existing methods for constructing ensembles of process-based models (i.e., bagging, boosting) and a single model, in terms of predictive performance. The results of the Friedman-Bonferroni-Dunn test are depicted by average rank diagrams, where the critical distance is shown as a dashed blue line.

Part III

Results

In this part we present the results of the empirical analysis of the methodology for learning ensembles of process-based models. The evaluation follows the experimental design presented in the previous chapter. In general, it serves both to find the optimal design choices and to illustrate the utility of the different algorithms for learning ensembles of process-based models.

In order to provide a detailed evaluation we perform two series of experiments for each algorithm of learning ensembles. In the former, we perform a comparative analysis of using (1) different methods for selecting the ensemble constituents that will be included in the ensemble, (2) different methods for combining the simulations of the ensemble constituents in the ensemble, and (3) different number of constituents in the ensemble. Based on the results of this comparative analysis, we make a set of (optimal) choices that we use for learning ensembles of process-based models. In the latter, we aim at analyzing the predictive performance of ensembles of process-based models. In particular, we test the central hypothesis of this thesis that ensembles of process-based models yield a predictive improvement compared to a single process-based model.

The ensemble framework is evaluated on predictive tasks of modeling population dynamics in aquatic ecosystems. The tasks include predictive modeling of phytoplankton concentration in three real-world lake domains of Lake Bled in Slovenia, Lake Kasumigaura in Japan and Lake Zurich in Switzerland. An overview of the real-world data is outlined in Chapter 4, whereas the complete library used for modeling aquatic ecosystems is given in Appendix A.

Given the fact that in this thesis we devolved several algorithms for learning homogeneous ensembles of process-based models, in the following sections we present the results based on the type of the ensembles learned. Chapter 5 presents the results of the empirical evaluation performed with ensembles learned with sampling data instances. Next, Chapter 6 outlines the results of the analysis performed for ensembles learned by sampling the library of domain knowledge. The results of the experiments performed with ensembles which are learned both by sampling the data and sampling the library of domain knowledge are presented in Chapter 7. In the last series of experiments, i.e. Chapter 7.3, we investigate whether the performance improvement is correlated to the diversity of the predictions of the ensemble constituents.

Note that some of the results in this thesis were already published in peer-reviewed journals. For that reason, this chapter embodies verbatim copies of the studies:

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015a). Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems. *Ecological Modelling*, *306*, 305–317.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015b). Predicting long-term population dynamics with bagging and boosting of process-based models. *Expert Systems with Applications*, *42*(22), 8484–8496.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2016). Modeling dynamic systems with efficient ensembles of process-based models. *PLoS ONE*, *11*(4), 1–27.

More specifically, the proof-of-concept study about the validity of the ensembles of process-based models, in particular bagging of process-based models, is presented in the paper titled *"Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems"* in Chapter 5. Next, the details of both bagging and boosting algorithms in terms of their methodology are presented in the paper, provided also in Chapter 5, titled *"Predicting long-term population dynamics with bagging and boosting of process-based models"*. Finally, the method for learning ensembles from random library subsamples together with the complete empirical analysis is presented in Chapter 6, in the publication titled *"Modeling dynamic systems with efficient ensembles of process-based models"*.

Summary of the Results

The results of the experiments can be summarized as follows. When learning ensembles of process-based models, one should use a separate validation data set in addition to the training one when learning the base-models included in the ensemble. This yields more robust ensembles, which in turn substantially improve the predictive performance. For combining the simulation of constituent process-based models, one should use the simplest combining scheme, i.e., averaging. The optimal ensembles of PBMs consist of a relatively small number of constituent models, ranging from 10 for the RLS method to 25 for the rest of the ensemble methods: bagging, boosting and BRLS. The process-based models, when simulated in a predictive setting, often produce divergent simulations, i.e., simulations where the systems variables leave their plausible ranges. Therefore, when simulating ensembles of PBMs, one should explicitly handle this kind of behavior of the base models. The proposed method in this thesis uses the provided domain knowledge on system variable ranges to discard the invalid behaviors from the resulting ensemble prediction. This can be viewed as a dynamic form of ensemble simulation.

The evaluation of the predictive performances of the single models and the four ensembles of PBMs learned is performed on predictive modeling tasks of modeling population dynamics in three real-world lakes. The results confirm the superiority of ensembles to a single model. The ensemble methods are far more robust than single models, which severely under-perform in several cases. In cases where single models outperform ensembles, the difference in performance when compared with the top-ranked ensemble method for the particular case is minor. On the other hand, the proposed ensemble methods cannot be distinguished in terms of performance. The differences in performances between the models obtained with the different ensemble methods are often minor and negligible.

In order to compare the proposed ensemble methods, we need to relate them to three criteria: average predictive performance, robustness and computational efficiency. Based on the first criterion, the winner is the *BRLS* method. The results of the average ranks show that the BRLS overall outperforms the other three methods, more specifically in 6 out of 15 cases. However, the BRLS method also severely underperforms compared to the single model in one case. Based on the second criterion, the *boosting* method is the most robust. Overall, boosting ensembles exhibit a stable behavior, which translates to predictive improvement in performance over the single model in cases when the other ensemble methods fail. Finally, the comparative analysis of the computational performances reveals that RLS ensembles are learned in a time comparable to the time needed to learn a single model. This is orders of magnitude faster when compared to bagging (nearly 25 times slower), boosting (nearly 25 times slower), and BRLS (nearly 15 slower) counterparts. This is a non-trivial advantage in computational efficiency as compared to the other methods for learning ensembles of process-based models.

Finally, we observed a varying degree of diversity between ensemble constituents for different data sets. Moreover, the measured correlation coefficients did not show any significant linear and/or monotonic relationship between the predictive improvement in performance and the intra-ensemble diversity.

Chapter 5

Learning Ensembles via Sampling Data Instances

In this chapter, we present the results of the empirical evaluation concerned with learning ensembles by sampling the data instances. We first explore the results obtained by predictive modeling phytoplankton concentration in real-world lakes. These results are already published in two journal publications, attached in the remainder of this chapter.

The first study in this chapter titled *"Learning ensembles of population dynamics and their application to modelling aquatic ecosystems"*, addresses the question whether ensembles of process-based models can be employed for predictive modeling tasks of dynamic systems. More specifically, this proof-of-principle study proposes that an ensemble method such as bagging, when applied in the context of process-based modeling for tasks of modeling dynamic systems, can yield to improved long-term predictive performance.

The ensemble method is evaluated on a set of predictive tasks of modeling population dynamics in aquatic ecosystems. Data on three lake ecosystems are used, together with a library of process-based knowledge on modeling population dynamics. Based on the evaluation results, the aim is to identify the optimal settings of the method for learning ensembles of process-based models, i.e., the optimal number of ensemble constituents, as well as the optimal way to select and combine them. Moreover, the aim is to investigate whether the ensembles of process-based models can accurately simulate the current and predict the future behavior of the three aquatic ecosystems.

The results of the experiments can be summarized as follows. When learning ensembles of process-based models using bagging, one should use a separate validation data set in addition to the training one when learning the base-models included in the ensemble. For combining the simulation of constituent process-based models, one should use the simplest combining scheme, i.e., averaging. The optimal ensembles of PBMs consist of a relatively low number of constituent models, ranging between 10 and 25. The process-based models, when simulated in a predictive setting, can often produce divergent simulations, i.e., simulations where the systems variables leave their plausible ranges. Therefore, when simulating ensembles of PBMs, one should explicitly handle this kind of behavior of the base models. To this end, we use the provided domain knowledge on system variable ranges to discard the invalid behaviors from the resulting ensemble prediction. This can be viewed as a dynamic (domain-knowledge based) form of ensemble pruning.

Next, the ensembles of 25 base-models selected using a validation data set and combined with the average combining scheme significantly outperform single models of population dynamics in aquatic ecosystems. The improvement of performance between the ensembles and the single models is positively related to the diversity of the ensemble constituents — the higher the diversity, the greater the improvement. However, the correlation between the diversity and the performance gain is weak and negligible, as a consequence of modest diversity between the ensemble constituents. Finally, the simulations of the ensembles show that ensembles are applicable for both predictive modeling tasks, where prediction of the future system behavior (test data error estimates) is of central interest, as well as descriptive modeling tasks, where the focus is on explaining the observed behavior (training data error estimates).

The second study titled *"Predicting long-term dynamics with bagging and boosting of process-based models"*, extends the limited proof-of-principle scope of the previous study, where a single ensemble method of bagging process-based models is employed, towards proposing a general methodology for learning ensembles of process-based models by sampling the data instances. Motivated by the already presented limitations of the state-of-the-art paradigm of process-based modeling, that is, process-based models have a limited ability to accurately predict the future behavior of an observed system, we implement two approaches of learning ensembles, i.e, bagging and boosting of process-based models. In this study, we perform an empirical evaluation of the implemented methods on three real-world modeling problems from the domain of population dynamics in aquatic ecosystems. The results of the empirical evaluation again confirms that ensembles of process-based models can lead to long-term predictions of the population dynamics that are more accurate than the ones obtained with a single process-based model.

More specifically, the results presented in this study empirically confirm that ensembles of process-based models yield a significant gain in predictive performance when compared to single models. Based on the empirical evaluation, we identified the main design decisions that need to be made when learning such ensembles by using bagging and boosting as underlying methods. In this context, it is important that one uses a separate validation data set in addition to the training one when learning the base models included in the ensemble. The optimal ensembles of PBMs consist of relatively low numbers of constituent models, ranging between 10 and 25 for bagging and 25 to 50 for boosting (for both methods, the best performing ensembles comprised 25 constituents). For combining the simulation of the constituent process-based models, the results showed that the simplest combining scheme, i.e averaging, provides the most satisfying results both in terms of predictive accuracy and computational complexity. Note that these findings are in line with our previous conclusions derived from the previous study.

Finally, our major conclusion is that both methods for learning ensembles of process-based models, following the optimal design decisions outlined above, outperform single process-based models. More importantly, bagged process-based models provide a significant performance gain over a single model.

Ecological Modelling 306 (2015) 305–317

Contents lists available at ScienceDirect

Ecological Modelling

journal homepage: www.elsevier.com/locate/ecolmodel

Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems

Nikola Simidjievski[a,b,*], Ljupčo Todorovski[c], Sašo Džeroski[a,b]

[a] *Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia*
[b] *Jožef Stefan International Postgraduate School, Ljubljana, Slovenia*
[c] *Faculty of Administration, University of Ljubljana, Slovenia*

ARTICLE INFO

Article history:
Available online 5 September 2014

Keywords:
Aquatic ecosystems
Population dynamics
Phytoplankton growth
Process-based modelling
Ensembles
Bagging

ABSTRACT

Ensemble methods are machine learning methods that construct a set of models and combine their outputs into a single prediction. The models within an ensemble can have different structure and parameters and make diverse predictions. Ensembles achieve high predictive performance, benefiting from the diversity of the individual models and outperforming them.

In this paper, we develop a novel method for learning ensembles of process-based models. We build upon existing approaches to learning process-based models of dynamic systems from observational data, which integrates the theoretical and empirical paradigms for modelling dynamic systems. In addition to observed data, process-based modelling takes into account domain-specific modelling knowledge.

We apply the newly developed method and evaluate its utility on a set of problems of modelling population dynamics in aquatic ecosystems. Data on three lake ecosystems are used, together with a library of process-based knowledge on modelling population dynamics. Based on the evaluation results, we identify the optimal settings of the method for learning ensembles of process-based models, i.e., the optimal number of ensemble constituents (25) as well as the optimal way to select (using a separate validation set) and combine them (using simple average). Furthermore, the evaluation results show that ensemble models have significantly better predictive performance than single models. Finally, the ensembles of process-based models accurately simulate the current and predict the future behaviour of the three aquatic ecosystems.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Mathematical models are widely used to describe the structure and predict the behaviour of dynamic systems under various conditions. Constructing such a model is a process that uses both expert knowledge and measured data about the observed system. The main challenge is integrating these two into an understandable model within the laws of nature.

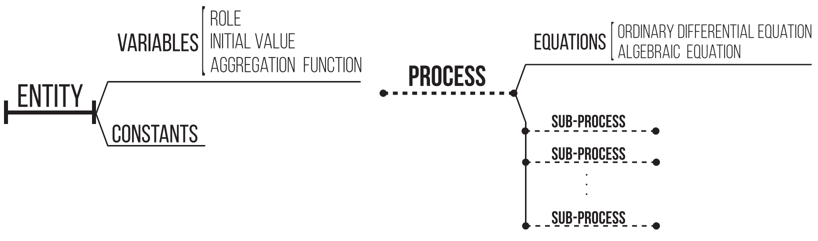
Two major paradigms for constructing models of dynamic systems exist: theoretical (knowledge-driven) and empirical (data-driven) modelling. Following the first paradigm, domain experts establish an appropriate structure of the model and calibrate its parameters in an automatic fashion using measured data. The second approach uses measured data to search for such a combination of model structure and parameter values that leads to simulated behaviour that fits the measurements well. In both approaches, the models are often formulated as ordinary differential equations (ODEs).

Within the area of computational scientific discovery (Langley et al., 1987), a sub-field of equation discovery has emerged that studies methods for learning the model structure and parameter values of dynamic systems from observations (Džeroski and Todorovski, 2003; Bridewell et al., 2008). The state-of-the-art approaches in this area, referred to as process-based modelling (Bridewell et al., 2008; Čerepnalkoski et al., 2012), integrate the theoretical and the empirical paradigm to modelling dynamic systems. A process-based model (PBM) provides an abstraction of the observed system at two levels: qualitative and quantitative.

At the qualitative level, a process-based model comprises entities and processes. Entities correspond to agents involved in the modelled system, whereas processes represent the relations and interactions between the entities. This results in an interpretable model of a system, explaining the structure of the observed system. On the other hand, at the quantitative level, the entities define a set

* Corresponding author at: Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia. Tel.: +386 1 477 3635.
E-mail addresses: nikola.simidjievski@ijs.si (N. Simidjievski), ljupco.todorovski@fu.uni-lj.si (L. Todorovski), saso.dzeroski@ijs.si (S. Džeroski).

http://dx.doi.org/10.1016/j.ecolmodel.2014.08.019
0304-3800/© 2014 Elsevier B.V. All rights reserved.



(a) Entity (b) Process

Fig. 1. The internal structure of entities and processes in process-based models.

of variables and constants, and the processes are annotated with equations modelling the underlying relations and interactions. At this level, we can transform a process-based model to a system of ODEs and simulate its behaviour.

Following the process-based modelling approach, we can generalize specific entities and processes into template entities and processes in a given modelling domain. A collection of such template entities and processes is called a library of process-based domain-specific knowledge. In modelling aquatic ecosystems, such a library of model components has been proposed by Atanasova et al. (2006b). The library defines a set of template entities, i.e., nutrients, primary producers, animals and environment, that typically occur in aquatic ecosystems (Luenberger, 1979). These entity templates are used to define template processes that provide recipes for modelling food-web interactions between the aquatic ecosystem entities. The knowledge encoded within the template entities and processes allow for automated modelling of population dynamics in aquatic ecosystems from measurements of system states (e.g., nutrients and species concentrations) through time. Process-based modelling software can then integrate the encoded knowledge with the measured system behaviour into a PBM of the observed system.

In our previous work, we have shown the utility of the process-based modelling approach for modelling population dynamics in a number of natural lakes (Čerepnalkoski et al., 2012) and marine ecosystems (Bridewell et al., 2008). Note however, that these studies focused on establishing descriptive, explanatory models of the population dynamics in aquatic ecosystems and the obtained models were analyzed and simulated on the same data that were used for learning them. In particular, they aimed to identify the limiting factors of the phytoplankton growth in the observed systems that are evident from the qualitative level of the learned process-based models. The generalization power of the obtained process-based models in terms of their ability to predict the future behaviour of the observed systems was not investigated in these studies.

In this study, we shift our focus towards the predictive performance of process-based models. The results of the preliminary experiments indicate the tendency of process-based models to overfit: While focusing on the provision of detailed and accurate descriptions of the observed systems, PBMs fail to accurately predict future system behaviour. To address this limitation of process-based models, we propose here a standard method for improving the predictive performance of models in machine learning, the use of ensembles. The idea of ensembles is to learn a set of predictive models (instead of a single one) and then combine their predictions. The prediction obtained with the ensemble is expected to be more accurate than the one obtained with a single model (Maclin and Opitz, 1999; Rokach, 2010).

The main contribution of this paper is a novel method for learning ensembles of process-based models. For tasks such as modelling the behaviour of ecosystems, the ensembles are usually employed in the context of learning tasks for classification and regression (Crisci et al., 2012; Knudby et al., 2010). However, to the best of our knowledge ensembles of process-based models have not yet been addressed in this context, and considered for tasks for modelling ecosystems.

We test the utility of the newly developed method for predictive modelling of population dynamics in lakes. To this end, we conjecture that ensembles of PBMs, similarly to other types of ensembles in machine learning, will improve the predictive performance of single models and lead to satisfactory prediction of future behaviour of the observed aquatic ecosystems. To test this hypothesis, we experiment on a series of tasks of modelling population dynamics in three lakes: Lake Bled, Lake Kasumigaura and Lake Zurich. From each lake we use seven yearly data sets, using six for learning and one for testing the predictive performance of the learned models. The aim of the experiments is two fold: Beside validating our central hypothesis (that ensembles perform better than single models), we also seek appropriate design choices related to our method for building ensembles of process-based models.

The remainder of this paper is organized as follows. Section 2 introduces the novel approach to learning ensembles of process-based models by discussing the task of automated modelling of dynamic systems – the process-based modelling approach, and focuses on a recent contribution to the area of automated process modelling, i.e., the ProBMoT tool. Section 3 describes ensemble methods in general and their adaptation for process-based modelling in particular. The design of the experiments, the evaluation measures and the data sets used are described in Section 4. Section 5 presents the results obtained the experimental evaluation. In Section 6, we discuss the contributions of this paper and overview the related work. Finally, Section 7 summarizes the work presented in this paper and discusses directions for further work.

2. Process-based modelling and ProBMoT

Equation discovery is the area of machine learning that aims at developing methods for learning quantitative laws, expressed in the form of equations, from collections of observed data. Recently, equation discovery methods have been used in the context of learning models of dynamic systems (Todorovski and Džeroski, 2007; Džeroski and Todorovski, 1993). The state-of-the-art equation discovery methods for modelling dynamic systems, referred to as process-based modelling (Bridewell et al., 2008; Džeroski and Todorovski, 2003) integrate domain-specific modelling knowledge and data into explanatory models of the observed systems. In the rest of this section, we briefly introduce the process-based modelling approach and then describe its particular implementation within the ProBMoT software platform.

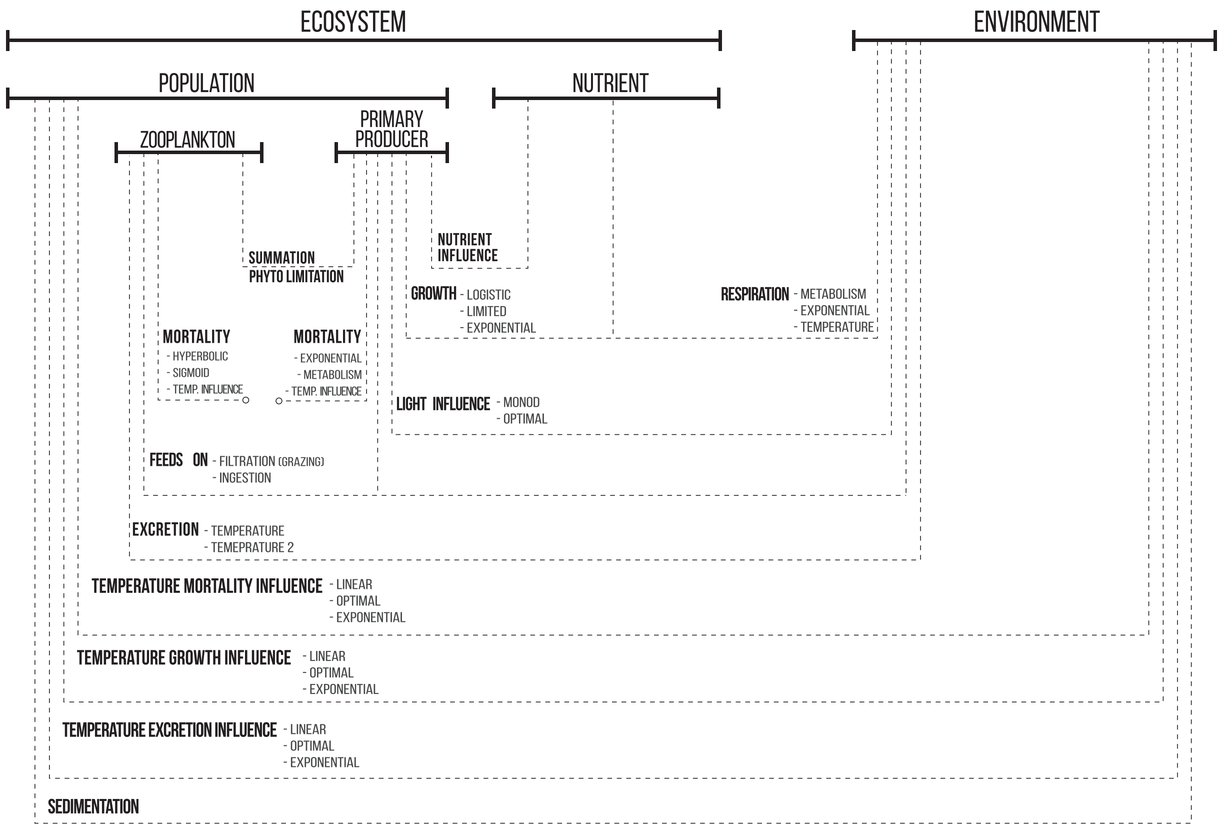

ECOSYSTEM
ENVIRONMENT
POPULATION
NUTRIENT
ZOOPLANKTON
PRIMARY PRODUCER
SUMMATION
PHYTO LIMITATION
NUTRIENT INFLUENCE
GROWTH - LOGISTIC - LIMITED - EXPONENTIAL
RESPIRATION - METABOLISM - EXPONENTIAL - TEMPERATURE
MORTALITY - HYPERBOLIC - SIGMOID - TEMP. INFLUENCE
MORTALITY - EXPONENTIAL - METABOLISM - TEMP. INFLUENCE
LIGHT INFLUENCE - MONOD - OPTIMAL
FEEDS ON - FILTRATION (GRAZING) - INGESTION
EXCRETION - TEMPERATURE - TEMEPRATURE 2
TEMPERATURE MORTALITY INFLUENCE - LINEAR - OPTIMAL - EXPONENTIAL
TEMPERATURE GROWTH INFLUENCE - LINEAR - OPTIMAL - EXPONENTIAL
TEMPERATURE EXCRETION INFLUENCE - LINEAR - OPTIMAL - EXPONENTIAL
SEDIMENTATION

Fig. 2. The library of modelling knowledge comprising template entities (thick horizontal lines) and processes (dashed lines connecting the entities) for modelling population dynamics in aquatic ecosystems.

2.1. Process-based modelling

Process-based models provide a description of the observed system at two levels of abstraction. At the upper, a qualitative level, process-based model consists of entities and processes. The entities represent the main components of the observed system, whereas the processes correspond to the interactions between the system components. At the qualitative level, process-based models provide insight into the high-level conceptual structure of the system. However, this high-level description does not provide enough details that would allow for simulation of the system behaviour.

On the other hand, at the quantitative level, entities and processes provide further modelling details that allow for the transformation of PBMs to ODEs and therefore simulation of the system. Fig. 1 depicts the internal structure of entities and processes, which defines a number of properties as follows.

Entities comprise variables and constants related to the components of the observed system. For example, an entity representing phytoplankton in an aquatic ecosystem would include a variable corresponding to its concentration, that changes through time, and a constant, corresponding to its maximal growth rate. Each entity variable has three important properties: the role in the model, the initial value and the aggregation function. The role of the variable in the model can be endogenous, i.e., representing internal system state, or exogenous, i.e., representing an input external to the system (not modelled within the system). An example of an endogenous variable in an aquatic ecosystem is the concentration of phytoplankton, while the water temperature is often treated as exogenous. Initial values of endogenous variables are necessary for model simulation. Moreover, each endogenous variable has its constraints defined, which limit the set of feasible values of the variable (for example, the concentration of the phytoplankton can neither be negative nor exceed 100 gWM/m^3). Finally the aggregation function for a variable specifies how influences from multiple processes on the specific variable are need to be combined, e.g., additively or multiplicatively.

The processes include specifications of the entities that interact, equations, and sub-processes. Consider the process of phytoplankton growth. It involves the phytoplankton as well as the growth limiting factors of nutrients and the environment. Equations provide the model of the interaction represented by the process and contains variables and constants from the entities involved in the corresponding interaction. In the phytoplankton growth example, an equation would define the mathematical model for calculating the growth rate. Finally, each process can include a number of sub-processes related to different aspects of the interaction. For example, the mathematical term of temperature limitation of growth (or nitrogen/nutrient), can be specified in an appropriate temperature (or nitrogen) limitation sub-process of the growth process. Sub-processes improve both the interpretability and the modularity of process-based models (Bridewell et al., 2008).

The entities and processes represent specific components and interactions observed in the particular system at hand. The process-based modelling approach allows for a higher-level representation of domain-specific modelling knowledge, employing the concepts of entity and process templates. They both provide general modelling recipes that can be instantiated to any specific components or interactions in the system. The phytoplankton entity from the example above is an instance of the general template entity of primary producer. Similarly, particular model of phytoplankton

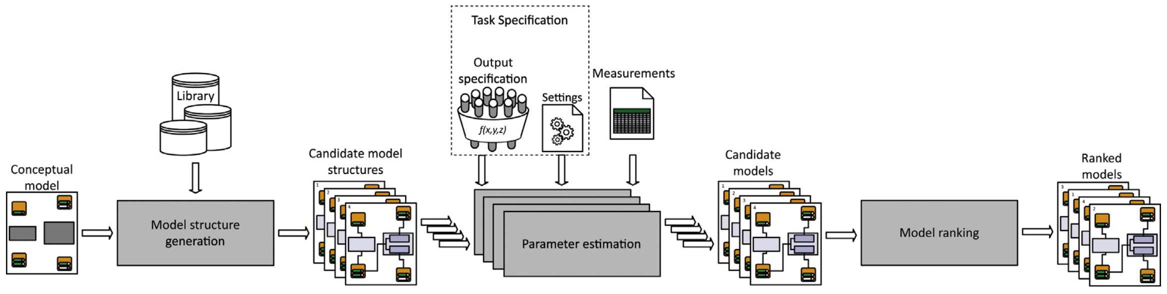


Fig. 3. The architecture of the ProBMoT platform for process-based modelling.

growth used in the above example process, is an instance of the more general growth process template. The template entities and processes are collected together into a library of components for modelling systems in a given domain of use.

Fig. 2 represents a high-level overview of the library for modelling population dynamics in aquatic ecosystems proposed by Atanasova et al. (2006b). The library organizes the templates in hierarchies. The hierarchy of entity templates (represented by thick horizontal lines in Fig. 2) in aquatic ecosystems includes the ecosystem entity and the environment entity templates at the highest level. The ecosystem entity template instantiates further on into population and nutrient entity templates, the first being further specialized into zooplankton and primary producer entity templates. Similarly, the process templates (represented by dashed lines, connecting involved entity templates) are organized into a hierarchy that defines the space of modelling alternatives. For example, the growth of a primary producer can be logistic, exponential or limited.

When learning process-based models, the entity and process templates from the library are instantiated to specific entities and processes corresponding to the observed system. These specific entities and processes represent model components that can be in turn used to define the set of candidate model structures. The algorithm for learning models employs knowledge-based methods from artificial intelligence to enumerate all candidate model structures. To evaluate a structure, the learning algorithm performs parameter estimations fitting the values of the constant model parameters that minimize the model error, i.e., the discrepancy between the model simulation and the observed system behaviour. The parameter estimation employs non-linear optimization to minimize the model error. Finally, the obtained models are sorted by decreasing model error and the best-ranked model is considered to be the result of the learning process.

Basic automated modelling algorithms perform exhaustive search through a constrained space of candidate process-based models, limiting the number of processes in the model (Bridewell et al., 2008). Advanced learning algorithms, such as Lagramge2.0 (Todorovski and Džeroski, 2007) and HIPM (Hierarchical Inductive Process Modelling) (Todorovski et al., 2005), perform heuristic search and allow for more sophisticated hierarchical constraints on the plausible process combinations. In the remainder of this section we will briefly describe the most recent implementation of the PBM approach, called ProBMoT, which stands for Process-Based Modelling Tool (Čerepnalkoski et al., 2012).

2.2. ProBMoT

Fig. 3 represents the architecture of the ProBMoT software platform for process-based modelling. ProBMoT supports the simulation, parameter estimation and automated learning of process-based models. ProBMoT follows the PBM approach described above.

The first input to ProBMoT is the conceptual model of the observed system. The conceptual model specifies the expected logical structure of the expected model in terms of entities and processes that we observe in the system at hand. ProBMoT combines the conceptual model with the library of modelling choices to obtain a list of candidate model structures. For each model structure, the parameter values are estimated to fit the observed behaviour of the modelled system.

The parameter estimation process is based on the meta-heuristic optimization framework jMetal 4.4 (Durillo and Nebro, 2011) that implements a number of global optimization algorithms. In particular, ProBMoT uses the Differential Evolution (DE) (Storn and Price, 1997) optimization algorithm. For simulation purposes, each process-based model is first transformed to a system of ODEs. In turn, ProBMoT employs the CVODE (C-package for Variable-Coefficient ODE) solver from the SUNDIALS suite (Cohen and Hindmarsh, 1996).

ProBMoT implements a number of measures of model performance: the sum of square errors (SSE) between the simulated and observed behaviour, and several variants thereof. The latter include mean squared error (MSE), root mean squared error (RMSE), relative root mean squared error (ReRMSE) and weighted root mean squared error (WRMSE). The last two are used in the experiments presented here, and will be explained in greater detail later, together with the particular ProBMoT parameter settings.

3. Ensemble methods and ensembles of process-based models

Learning ensembles is an established method for improving the predictive performance of models in machine learning (Okun et al., 2011), however learning ensembles of process-based models has not been considered so far. In this section, we define ensembles of process-based models and corresponding methods for learning them. First, however, we provide a brief overview of classical ensemble methods in machine learning.

An ensemble is a set of models (referred to as base-models or ensemble constituents) that is expected to lead to predictive performance gain over a single model. The idea behind ensembles is to improve the overall predictive power by combing the predictions of individual base-models. An ensemble method consists of three main components: a technique for learning/generating a set of candidate base-models, a technique for selecting the base-models that constitute the ensemble, and a combining scheme specifying how the base-model predictions are aggregated into an ensemble prediction.

Based on how the candidate base-models are learned, the ensembles can be homogeneous or heterogeneous. In homogeneous ensembles, the base-models are learned with the same learning algorithm, but from different samples of the training data. Commonly used sampling variants include: sampling of data instances (bagging Breiman, 1996a, boosting Freud and Schapire, 1999), sampling of data features/attributes (random subspaces Ho, 1998) or both (random forests Breiman, 2001). On the other hand, in heterogeneous ensembles, the candidate base-models are learned using different learning algorithms (e.g., stacking Wolpert, 1992).

After we have generated the candidate base-models, we have to select the ones to be included in the ensemble. Most classical ensemble methods would typically use all the candidate models as ensemble constituents. In contrast, ensemble pruning techniques can be used to learn small-size ensembles (thus reducing the computational complexity) and improve ensemble robustness, e.g., in the case of bagging (Zhou et al., 2002).

Finally, the combining scheme depends on the type of the base-models. In the case of classification models that predict qualitative values, different voting scheme are employed. In the case of regression models that predict numeric values, the alternatives include average, weighted average and weighted median (Drucker, 1997).

In this paper, we adapt the well-known bagging method for learning homogeneous ensembles where the training data is modified by sampling the data instances. The *bagging* method, introduced by Breiman (1996a), is one of the earliest and simplest ensemble learning methods. It first randomly samples data instances, with replacement, to obtain several bootstrap replicates of the training data. Next, a candidate base model is learned from each of the different bootstrap replicates. An important property of bagging is that it can be implemented as a parallel algorithm, which is due to the fact that it handles each bootstrap sample independently.

In the continuation of this section, we introduce a novel approach to learning ensembles of process-based models. This approach follows the bagging idea introduced above. We are going to introduce it following the three-components structure of ensemble methods as outlined above.

3.1. *Learning individual process-based models*

Using ProBMoT, we learn the individual candidate base-models from different samples of the observed behaviour at hand. The notable difference from bagging in the context of regression is that, in our case, the data instances have temporal ordering that has to be preserved in each data sample. To achieve this, we implement the sampling by introducing weights for each instance. The weight corresponds to the number of times the instance has been selected in the process of sampling with replacement. Instances that have not been selected (the ones with weight 0) are simply omitted from the sample. From each sample, a PBM is learned with ProBMoT.

To account for the instance weights when learning a model from the sample, we employ the weighted root mean squared error in ProBMoT:

$$WRMSE(m) = \sqrt{\frac{\sum_{t=0}^{n} \omega_t * (y_t - \hat{y}_t)^2}{\sum_{t=0}^{n} \omega_t}}. \quad (1)$$

Here y_t and $\hat{y}_t$ correspond to the *measured* and *simulated* values (simulating the base model m) of the system variable y at time point t. n denotes the total number of instances in the data sample and ω_t denotes the weight of the data instance at time point t.

3.2. *Selecting and combining process-based models into an ensemble*

When learning a PBM from each data sample, ProBMoT selects the top-ranked model as a result. However, we can use two alternative data sets to calculate the error used to rank the models in ProBMoT. By default, ProBMoT ranks the models using the error on the training data sample; we refer to this selection method as *regular*. In contrast, the *validation* selection method employs a separate validation data set to calculate the error used for model ranking in ProBMoT.

In order to simulate an ensemble, we need to simulate every candidate base model. The resulting ensemble simulation is a combination of the predictions of all individual base-models in the respective time point. In our case, we use average, weighted average and weighted median as combining schemes: These are commonly used for tasks such as regression (Drucker, 1997). In the case of average, all base-models participate in the resulting simulation equivalently. For the weighted average and weighted median schemes a confidence is calculated for each of the base-models with respect to their performance error. The base-models with higher confidence will dominate in the resulting ensemble simulation.

However, some of these simulations may not be valid, i.e., may not satisfy the constraints given in the library of background knowledge. In this case, we perform ensemble pruning, i.e, we discard these base-models from the resulting ensemble. Bellow, we illustrate the necessity of using ensemble pruning by comparing two ensembles with 100 base-models.

Table 1
Predictive performance (ReRMSE on the testing data) errors of the complete and pruned ensembles and the number of base-models pruned from the 100 model ensembles learned on 15 data sets, described in see Section 4.3.

Case	Complete	Pruned	# base-models pruned
B1	1.8E+03	1.055	53
B2	1.243	1.243	0
B3	17.037	1.046	11
B4	0.737	0.737	1
B5	0.625	0.625	0
K1	9.2E+01	0.927	19
K2	4.482	1.840	98
K3	4.0E+04	0.907	17
K4	0.823	0.988	9
K5	0.978	0.978	0
Z1	1.105	1.028	2
Z2	1.187	1.077	3
Z3	8.579	1.212	6
Z4	0.972	0.972	0
Z5	2.8E+01	1.390	24

Table 1 presents the results of the comparison of two ensembles, complete and pruned, and the number of candidate base-models discarded, in terms of performance error on the test data sample (see Section Section 4.3) for 15 different cases. From the table, we can see that in all but one experimental data set (K4), the pruned ensemble outperforms (or is equal in performance to) the complete ensemble. Note also that, in several cases (B1, B3, K1, K3, Z5) the performance of the ensemble is significantly improved. By discarding the base-models with unstable simulations from the ensemble, we can ensure valid ensemble prediction and a stable simulation. In this paper, we use ensemble pruning of this kind as a standard technique when selecting the ensemble constituents and simulating the ensemble prediction.

4. Experimental setup

In this section, we present the setup of the experiments we performed to empirically evaluate the performance of the method for learning ensembles of process-based models. We perform the evaluation on tasks of modelling population dynamics in three aquatic ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan, and Lake Zurich in Switzerland. The goal of our empirical evaluation is twofold.

First, we are looking for a set of optimal design decisions related to the algorithm for learning ensembles. In particular, we want

Table 2
Overview of the data used for modelling population dynamics in the three lakes: Lake Bled, Lake Kasumigaura and Lake Zurich.

	Bled	Kasumigaura	Zurich
Environmental influence	Temperature Light	Temperature Light	Temperature Light
Nutrients	Phosphorus Nitrogen Silica	Phosphorus Nitrogen Ammonia	Phosphorus Nitrogen Silica
Primary producer	Phytoplankton	Phytoplankton	Phytoplankton
Zooplankton	*D. hyalina*	None	*D. hyalina*
Training data (labels)	1996–2000 (B1–B5)	1986–1990 (K1–K5)	1996–2000 (Z1–Z5)
Validation data	2001	1991	2001
Test data	2002	1992	2002

to perform a comparative analysis of using different methods for learning and the base-models to be included in the ensemble, different methods for combining the simulations of the base-models in the ensemble, and different numbers of based models in the ensemble. Based on the results of this comparative analysis, we make a set of choices that we use for learning ensembles of process-based models of aquatic ecosystems.

Second, we aim at analysing the predictive performance of ensembles of process-based models. In particular, we test the central hypothesis of this paper that ensembles of process-based models improve the predictive power of a single process-based model for a given aquatic ecosystem. Furthermore, we want to investigate whether the performance improvement is related to the diversity of the predictions of the ensemble constituents. Finally, in the last series of experiments, we visually compare the predictions of ensembles to those of single models in each of the three aquatic ecosystems.

4.1. *Library of domain-specific knowledge and task of modelling aquatic ecosystems*

In our experiments, we use the library of domain-specific knowledge for process-based modelling of aquatic ecosystems presented by Čerepnalkoski et al. (2012). Note that the library is based on the previous work presented by Atanasova et al. (2006b). The library, presented in Fig. 2, formalizes modelling knowledge in terms of a set of template entities and processes for modelling population dynamics in an arbitrary lake ecosystem. To reduce the computational complexity of the experiments performed in this paper, we used a simplified version of the library, where we omitted some of the alternatives for modelling individual processes. The simplified version of the library and the conceptual model, lead to 320 candidate model structure for Lake Kasumigarua and 128 candidates for the other two aquatic ecosystems used.

ProBMoT employs the Differential Evolution (DE, Storn and Price, 1997) method for parameter estimation with the following settings: population size of 50, *rand*/1/*bin* strategy, and the differential weight (F) and the crossover probability (Cr) both set to 0.6. The limit on the number of evaluations of the objective function is one thousand per parameter. For simulating the ODEs we used the CVODE simulator with absolute and relative tolerances set to 10^{-3}. The particular choice of parameters setting of DE is based on previous studies of the sensitivity of DE for estimating parameters of ODE models, which includes also modelling of the population dynamics in Lake Bled (Taškova et al., 2011, 2012).

4.2. *Data*

The data used in this study originates from three aquatic ecosystems: Lake Bled, Lake Kasumigaura and Lake Zurich.

Lake Bled is of glacial-tectonic origin, located in the Julian Alps in north-western Slovenia (46.3644° N, 14.0947° E). It occupies an area of 1.4 km^2, with a maximum depth of 30.1 m and an average depth of 17.9 m. The measurements, performed by the Slovenian Environment Agency, consist of physical, chemical and biological data for the period from 1996 to 2002. All the measurements were performed once a month and depth-averaged for the upper 10 m of the lake. To obtain daily approximations, the data was interpolated with a cubic spline algorithm and daily samples were taken from the interpolation (Atanasova et al., 2006c).

Lake Kasumigaura, is located 60 km to the north-east of Tokyo, Japan (36.0403° N, 140.3942° E). It has an average depth of 4 m, a volume of 662 million cubic metres, and a surface area of 220 km^2. The data set comprises monthly measurements in the period from 1986 to 1992. Again, to obtain daily approximations, the measurements were interpolated using linear interpolation and daily samples were taken from the interpolation (Atanasova et al., 2006a).

Lake Zurich is located in the south-western part of the canton of Zurich in Switzerland (42.1970° N, 88.0934° W). It has an average depth of 49 m, volume of 3.9 km^3 and a surface area of 88.66 km^2. The data comprises measurements performed by the Water Supply Authority of Zurich in the period from 1996 to 2002. The measurements, taken once a month, include profiles of physical, chemical and biological variables from 19 different sites. They were weight averaged to the respective epilimnion (upper ten metres) and hypilimnion (bottom ten metres) depths. The data was interpolated with a cubic spline algorithm and daily samples were taken from the interpolation (Dietzel et al., 2013).

We use the same structure of population dynamics model in all three aquatic ecosystems. It includes a single equation (ODE) for a system variable representing the phytoplankton biomass (measured as *chlorophyll-a* in Lake Kasumigaura). The exogenous variables include the concentration of zooplankton *Daphnia hyalina* (available only for Bled and Zurich), dissolved inorganic nutrients of nitrogen, phosphorus, and silica (ammonia in Lake Kasumigaura), as well as two input variables representing the environmental influence of water temperature and global solar radiation (light).

Table 2 provides a summary of the data sets we used in the experiments. For each aquatic ecosystem, we used seven data sets corresponding to the last seven years of available measurements. Five of these were used (one at a time) for training the base-models, one was used for validating the models in the process of selecting the ensemble constituents, and one was used to measure the predictive performance of the learned models and ensembles.

In each learning experiment, we take a single (year) training data set, learn a single model or an ensemble using the training and the validation data set, and test the predictive performance of the learned models on the test data set. We therefore perform 15 experiments. In the tables, we label them by the label of the training set used, which is comprised of the initial letter of the lake name, followed by a digit for 1 to 5 corresponding to each of the

Table 3
Comparison of the predictive performance (ReRMSE on test data) of the different combinations of methods for selecting (Validation and Regular) and methods for combining (Average, Weighted Average and Weighted Median) ensemble constituents applied to the 15 data sets. The numbers in bold are the best performance figures for the given data set.

	Validation			Regular		
	Average	Weighted average	Weighted median	Average	Weighted average	Weighted median
B1	1.08	1.09	1.08	**1.06**	1.09	**1.06**
B2	**1.10**	**1.10**	1.12	1.24	1.24	1.27
B3	0.97	0.97	**0.96**	1.05	1.04	1.05
B4	0.78	0.78	0.76	**0.74**	**0.74**	**0.74**
B5	0.87	0.88	0.87	0.63	0.63	**0.61**
K1	**0.73**	0.74	0.80	0.93	0.92	0.93
K2	1.64	**1.63**	1.64	1.84	1.80	1.60
K3	**0.87**	**0.87**	0.89	0.91	0.93	0.94
K4	**0.78**	**0.78**	0.79	0.99	0.99	0.98
K5	0.74	**0.73**	**0.73**	0.98	0.99	1.00
Z1	**0.78**	**0.78**	0.79	1.03	1.03	0.99
Z2	**0.88**	**0.88**	0.90	1.08	1.09	1.01
Z3	0.95	0.99	**0.91**	1.21	1.22	1.19
Z4	**0.96**	0.97	0.97	0.97	0.97	0.98
Z5	1.32	1.38	**1.26**	1.39	1.45	1.40

Table 4
Comparison of the descriptive performance (ReRMSE on training data) of the different combinations of methods for selecting (Validation and Regular) and methods for combining (Average, Weighted Average and Weighted Median) ensemble constituents applied to the 15 data sets. The numbers in bold are the best performance figures for the given data set.

Case	Validation			Regular		
	Average	Weighted average	Weighted median	Average	Weighted average	Weighted median
B1	0.27	0.36	0.27	**0.20**	0.29	0.21
B2	0.58	0.58	0.60	**0.24**	**0.24**	**0.24**
B3	0.37	0.40	0.41	**0.29**	**0.29**	**0.29**
B4	0.34	0.34	0.33	0.26	**0.25**	0.26
B5	0.56	0.56	0.56	**0.14**	**0.14**	**0.14**
K1	1.07	1.06	1.13	0.69	**0.68**	0.72
K2	0.83	0.84	0.84	**0.76**	**0.76**	**0.76**
K3	0.68	0.66	0.87	**0.54**	0.56	0.55
K4	0.68	0.74	0.84	**0.40**	**0.40**	**0.40**
K5	0.46	0.49	0.53	0.37	0.37	**0.38**
Z1	0.88	0.87	0.90	0.53	**0.52**	0.55
Z2	0.85	0.87	0.92	0.54	**0.53**	0.54
Z3	0.82	0.82	0.84	0.69	**0.68**	0.70
Z4	0.79	0.79	0.79	**0.77**	**0.77**	**0.77**
Z5	0.61	0.61	0.60	**0.51**	**0.51**	0.53

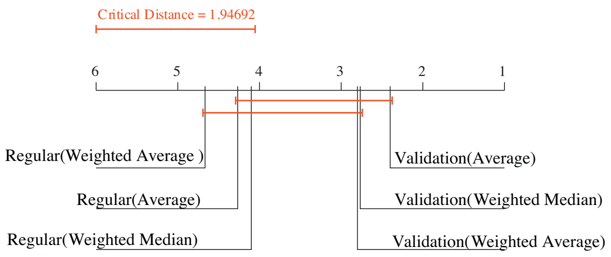


Fig. 4. Comparison of the average ranks of different combinations of methods for selecting and methods for combining ensemble constituents in terms of predictive performance (ReRMSE on test data) averaged over 15 data sets.

5 consecutive years of measurements. The labels are thus B1–B5, K1–K5 and Z1–Z5 and B5, for example, denotes the measurements for Lake Bled taken in the fifth year, i.e., the year 2000.

4.3. Evaluation methodology

To apprise the predictive performance of a given model m, we use the relative root mean squared error (*ReRMSE*) (Breiman, 1984), defined as:

$$ReRMSE(m) = \sqrt{\frac{\sum_{t=0}^{n}(y_t - \hat{y}_t)^2}{\sum_{t=0}^{n}(\bar{y} - \hat{y}_t)^2}}, \tag{2}$$

where n denotes the number of measurements in the test data set, y_t and $\hat{y}_t$ correspond to the *measured* and *predicted* value (obtained by simulating the model m) of the system variable y at time point t, and $\bar{y}$ denotes the mean value of the system variable y in the test data set. Note that the usual root mean squared error is normalized here with the standard deviation of the system variable in the test data, thus allowing us to compare the errors of models for system variables measured on different scales. The other interpretation of the normalization term is that it represents the error of a base-line model that predicts the average value of y at each time point t. Thus, the model with *ReRMSE* of 1 has performance equal to that of the base-line "average" predictor. Smaller values of *ReRMSE* indicate better predictive performance.

We observe the performance of different learning algorithms in terms of the predictive performance (*ReRMSE*) of the models learned on each of the 15 data sets. To assess the significance of the differences in performance between different learning algorithms, we use the corrected (Iman and Davenport, 1980) Friedman test (Friedman, 1940) and the post-hoc Nemenyi test (Nemenyi, 1963). This is a standard framework for comparing the predictive performance of different learning algorithms, superior to alternative frameworks as argued by Demšar (2006). The Friedman non-parametric test for multiple hypotheses testing first ranks the algorithms according to their performance (i.e., predictive performance of the trained models) on each combination of train/test data set, and then averages these ranks across all the data set

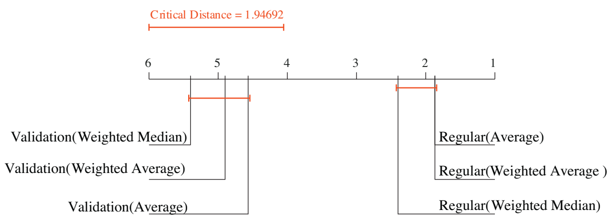


Fig. 5. Comparison of the average ranks of different combinations of methods for selecting and methods for combining ensemble constituents in terms of descriptive performance (ReRMSE on training data) averaged over 15 data sets.

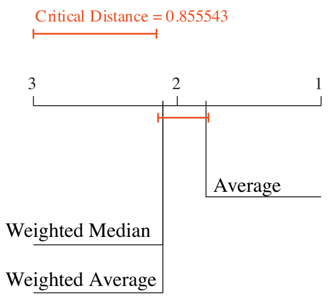


Fig. 6. Comparison of the average ranks of the three schemes for combining base-model simulations (Average, Weighted Average, and Weighted Median) in terms of predictive performance (averaged over 15 data sets). We compare the test errors of the ensembles with 100 base-models, selected by using a separate validation data set.

combinations. If the Friedman test indicates a statistically significant difference, we proceed with performing a Nemenyi test to identify which differences are significant.

The Nemenyi test computes the critical distance between the algorithm ranks at a given level of statistical significance (in our case, we set the significance level threshold at 95 %, $p = 0.05$). Only differences in the average ranks larger than the critical distance are considered significant; for those we can claim that one algorithm outperforms (i.e., performs significantly better than) the other. The results of the Friedman–Nemenyi tests are depicted by using average rank diagrams (Figs. 4–7). In these diagrams, we can see the name of each of the compared algorithms along with its average rank.

Finally, to test the conjecture that the power of ensembles is based on the exploitation of the diversity of the ensemble constituents, we measure the diversity of the ensemble constituents and correlate it to the performance improvement of ensembles over single models. To measure the diversity of the base-models in the ensemble e, we measure the average pairwise difference of the base model simulations

$$Diversity(e) = \frac{1}{\binom{|e|}{2}} \sum_{\{m_1, m_2\} \subset e} \sqrt{\frac{\sum_{t=0}^{n} (y_{1,t} - y_{2,t})^2}{n}}, \tag{3}$$

where $|e|$ denotes the number of base-models in the ensemble e, n the number of measurements in the data set, m_1 and m_2 two models from e, and $y_{1,t}$ and $y_{2,t}$ the simulated values of these models at time point t. To assess the performance improvement of the ensemble e over a single model m, we calculate

$$Improvement(e, m) = -\frac{ReRMSE(e) - ReRMSE(m)}{ReRMSE(m)}, \tag{4}$$

where the model m is learned from the complete training set, and the base-models from the ensemble e are learned on different bootstrap samples of the training set. We draw a scatter plot that depicts the correlation between ensemble diversity and performance improvement and calculate the Pearson Correlation Coefficient between them.

5. Results

In this section, we present and discuss the results of the empirical evaluation. We first explore some design decisions employed in the ensemble learning algorithm. We then analyze the improvement of performance obtained by replacing single models with ensembles and investigate the relation between the diversity of the ensemble constituents and the performance improvement. Finally, we visually compare the simulations of ensembles with the simulations of single models on both training and test data.

5.1. Selecting and combining ensemble constituents

One of the important decision when designing an algorithm for learning ensembles is how to select the models to be included in the ensemble. The base-line method (labelled *regular* in the tables and figures below) often employed in ensemble learning algorithms, is to select the models performing best on the bootstrap samples of the training data on which they were learned. Here, to avoid overfitting of the training data, we also consider an alternative method, labelled *validation* in the tables and figures below. We still build models on different bootstrap samples of the training data, but we evaluate their performance on a single separate validation set. In this first series of experiments, we construct ensembles of 100 base-models.

Table 5
Comparison of the predictive performance (ReRMSE on test data) of the single model and bagging ensembles that include 5, 10, 25, 50, and 100 base-models on the 15 data sets. The numbers in bold are the best performance figures for the given data set.

Case	single model	Ensemble 5	Ensemble 10	Ensemble 25	Ensemble 50	Ensemble 100
B1	1.22	**1.06**	**1.06**	1.07	1.09	1.08
B2	1.14	1.24	1.14	**1.09**	**1.09**	1.10
B3	1.07	1.03	0.99	**0.97**	**0.97**	**0.97**
B4	0.92	**0.75**	**0.75**	0.76	0.77	0.78
B5	0.92	0.87	0.89	**0.86**	0.87	0.87
K1	0.74	0.75	**0.73**	0.74	0.74	**0.73**
K2	2.20	1.50	**1.38**	1.43	1.52	1.64
K3	0.96	**0.86**	**0.86**	0.87	0.87	0.87
K4	0.78	0.77	**0.76**	0.77	0.78	0.78
K5	**0.72**	0.85	0.79	0.73	0.74	0.74
Z1	0.78	**0.77**	**0.77**	0.78	0.78	0.78
Z2	0.95	0.89	0.94	**0.88**	0.87	**0.88**
Z3	0.99	0.92	0.96	0.93	**0.92**	0.95
Z4	**0.94**	0.99	0.98	0.96	0.97	0.96
Z5	1.65	1.30	**1.11**	**1.11**	1.22	1.32

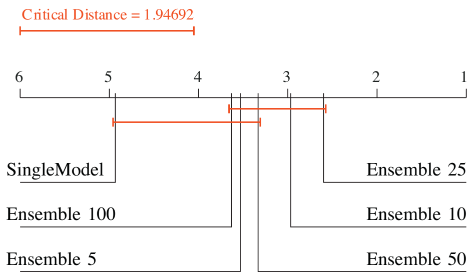


Fig. 7. Comparison of the average ranks of the single model and bagging ensembles that include 5, 10, 25, 50, and 100 base-models in terms of their predictive performance (ReRMSE on test data) averaged over the fifteen data sets.

Table 3 and Fig. 4 summarize the results of the comparison between the regular and the validation methods. From the table, we can see that for all but three data sets (B1, B4 and B5), the selection method based on a separate validation data set outperforms the regular base-line method. For four data sets (B3 and Z1–Z3), the use of validation lowered the ensemble *ReRMSE* below the value of 1, improving over the base-line “average” predictor. The Friedman–Nemenyi test (and the corresponding diagram in Fig. 4) confirms the significance of the observed superiority of the validation method: All methods that use validation are ranked better than those that use the training set, where the best method (validation used together with the average combining scheme) *significantly* outperforms the worst method (regular selection used with the weighted average combining scheme).

Note that we made the implicit conjecture that the regular selection method overfits the training data. The results presented in Table 4 and Fig. 5 confirm the validity of this conjecture. From the table, we can see that the regular method consistently leads to significantly smaller errors on the training data. Together with the results in Table 3/Fig. 4 these show a clear case of overfitting — while being superior on the training data, regular selection leads to inferior predictive performance as compared to validation-based selection.

Taken together, the above results show that the selection method based on validation is the right choice when learning ensembles of process-based models.

We next consider the choice of an appropriate method for combining the simulations/predictions of the models in the ensemble. Here, we choose among the three methods commonly used in learning ensembles of regression models: average, weighted average, and weighted median (Breiman, 1984; Drucker, 1997). Above, we considered these in combination with the regular and the validation selection methods: Here, we consider them in combination with the validation selection method only.

The results of the Friedman–Nemenyi test depicted in Fig. 6 show the lack of significant difference among the average ranks of the three combination schemes. However, the simplest among them, i.e., the ‘average’ method has also the best rank. Therefore, despite the lack of significant difference, we can make the decision to use ‘average’ as the most appropriate method for combining the predictions of the base-models in ensembles of PBMs.

In all the experiment performed so far, we learned ensembles of 100 base-models. However, the optimal number of base-models can depend on the type of the base-models in the ensemble. In the next series of experiments, we aim at identifying an optimal number of base-models to be included in the ensembles of PBMs for modelling aquatic ecosystems. To this end, we compare the predictive performance of ensembles consisting of 5, 10, 25, 50, and 100 models (learned on bootstrap samples) with the performance of a single model (learned on the complete data set).

Table 6
Diversity of the base-models and the relative improvement of the ensemble error over the error of the single model (given as percentage) for the fifteen data sets.

Case	Diversity	Relative improvement (%)
B1	0.354	12.27
B2	0.561	4.48
B3	0.230	9.24
B4	0.617	17.45
B5	0.270	6.81
K1	1.010	1.04
K2	1.030	35.06
K3	0.543	9.45
K4	0.598	1.02
K5	0.605	−1.53
Z1	0.089	0.69
Z2	0.234	7.23
Z3	0.223	5.72
Z4	0.125	−2.33
Z5	0.285	32.69
Pearson *r*	–	0.274

Table 5 and Fig. 7 summarize the results of these experiments. Comparing the predictive performance of thr different ensembles, we can see that for eight (out of fifteen) data sets, the ensemble of 10 base-models performs best, followed by the ensemble of 25 base-models, performing best for five data sets. The corresponding Friedman–Nemenyi diagram shows that the ensemble of 25 base-models is ranked best among all the ensembles. Note, however, that the critical distance on the same diagram shows that there is no significant difference in performance between any pair of ensembles with different numbers of constituents. Despite the lack of a significant difference, we are going to choose the ensembles with 25 base-models, which are ranked best, to be the subject of the further experiments.

When it comes to comparing the performance of the ensembles to that of a single model, the Friedman–Nemenyi test shows that the ensembles consisting of 10 and 25 models *significantly* outperform the single models. The tabular comparison of the predictive performance (Table 5) shows that, for all but two data sets (K5 and Z4), a single model performs worse than an ensemble. However, for these two data sets, the difference in the performance between the single model and an ensemble of 25 base-models is almost imperceptible.

These results clearly confirm the central hypothesis of this paper that the ensembles of PBMs significantly outperform a single PBM model. The optimal design choices to be used for learning ensembles are as follows: perform 25 iterations, in each of them select the best model with respect to the error measured on a separate validation set, and combine the ensemble constituents using the average combining scheme.

5.2. Ensemble diversity and performance improvement

The experimental results presented above show that ensembles of PBMs outperform single PBM models. Here, we further analyze the improvement and its relation to the diversity of the simulations of the ensemble constituents. To this end, we first measure the relative improvement of the performance obtained by using an ensemble as compared to using a single model. Then, we measure the diversity of base-models in the ensemble. Finally, we analyze the correlation between the two.

Table 6 and Fig. 8 summarize the results of these experiment. First, Table 6 confirm our previous finding: ensembles outperform single models in all but two data sets (K5 and Z4). Note that the loss of performance of the ensemble is minor (below 3%) for these two data sets. On the other hand, the gain in performance (performance improvement) can be substantial and reach up to 17% for Lake Bled (B4), 35% for Lake Kasumigaura (K2) and 33% for Lake Zurich (Z5).

Furthermore, we observe a varying degree of diversity between ensemble constituents for different data sets – diversity varies from 0.125 to 1.030. The scatter plot in Fig. 8 shows weak positive correlation between ensemble diversity and relative improvement of performance. The measured Pearson correlation coefficient of 0.274 is certainly neither high nor significant. Still, the positive correlation is in line with the ensemble literature assumption that ensembles perform well by exploiting the diversity of their constituents (Kuncheva and Whitaker, 2003).

5.3. Simulating ensembles

Finally, in the last series of experiments, we visually inspect the difference between the simulations of ensembles and simulation of single models. We selected one data set for each of the three aquatic ecosystems considered in the experiments and simulated the ensemble and the single model on both training and test data. The simulation on test data is in line with the predictive modelling setting used throughout the experiments presented in this paper. The simulation on the training data is in line with the previous work on building descriptive models of aquatic ecosystems, where only the performance on training data is considered (Čerepnalkoski et al., 2012; Taškova et al., 2012; Atanasova et al., 2006c).

In Fig. 9, we present the simulations of ensembles and single models in both the predictive (graphs on the left-hand side) and the descriptive scenario (graphs on right-hand side), for each of the three lakes. The first row presents the simulations for Lake Bled, the second for Lake Kasumigaura and the last for Lake Zurich. The visual comparison confirms the superiority of the ensembles in the predictive scenario. Note that only ensembles lead to acceptable reconstruction of the population dynamics for the test data sets. But more importantly, given the nature of the ensembles (that avoid overfitting), they do not seem to have lower descriptive performance; they still capture the population dynamics of the phytoplankton on training data. Thus, we can conclude that the ensembles can be applied in both predictive and descriptive scenarios.

One interesting case is Lake Zurich (Fig. 9e and f), where despite the high relative error (above 1), we can see that the ensemble accurately captures the phytoplankton dynamics with a slight phase shift.

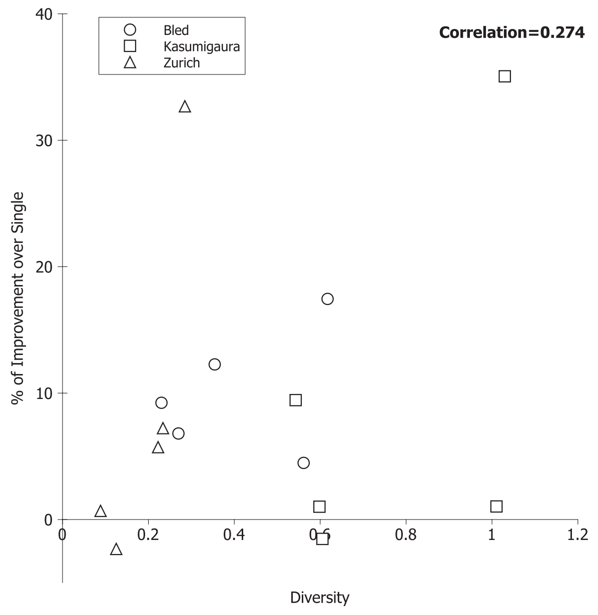


Fig. 8. Scatter plot depicting the correlation between the diversity of the base-model predictions and the relative improvement of error between an ensemble and a single model for the 15 data sets.

5.4. Summary

We can summarize the results of our experiments as follows. When learning ensembles of PBMs, one should use a separate validation data set in addition to the training one when learning the base-models included in the ensemble. For combining the simulation of constituent process-based models, one should use the simplest combining scheme, i.e., averaging. The optimal ensembles of PBMs consist of a relatively low number of constituent models, ranging between 10 and 25.

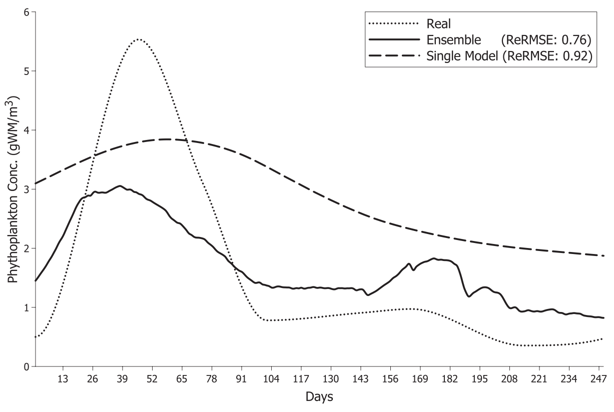
Furthermore, the ensembles of 25 base-models selected using a validation data set and combined with the average combining scheme significantly outperform single models of population dynamics in aquatic ecosystems. The improvement of performance between an ensemble and a single model is positively related to the diversity of the ensemble constituents – the higher the diversity, the greater the improvement. However, the correlation between the diversity and the performance gain is weak, as a consequence of the modest diversity between the base-models. Finally, the simulations show that ensembles are applicable for both predictive modelling tasks, where prediction of the future system behaviour (test data error estimates) is of central interest, as well as descriptive modelling tasks, where the focus is on explaining the observed behaviour (training data error estimates).

6. Discussion

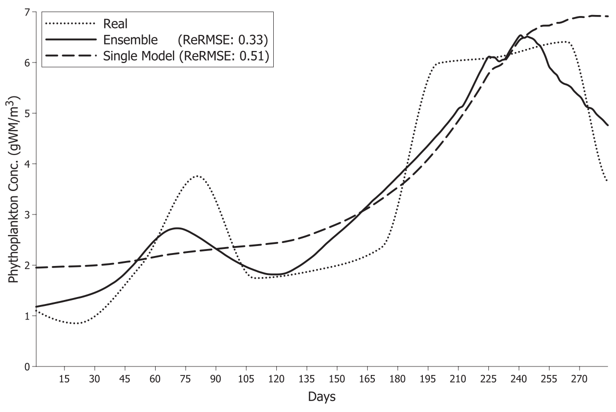
In this section, we discuss the method we propose and its results, and put them in the context of related work.

The work presented in this paper follows two different lines of research. First, it extends the state-of-the-art in the paradigm of equation discovery. More specifically, we build upon previous methods for learning process-based models, that have proven successful for automated modelling of population dynamics in a number of aquatic ecosystems (Todorovski and Džeroski, 2007; Čerepnalkoski et al., 2012; Bridewell et al., 2008). Second, it follows the basic principles of ensemble learning, and translates them into a methodology for modelling dynamic systems. Our work is closely related to that of Bridewell et al. (2005), where the authors use ensemble methods to establish better descriptive models by tackling the over-fitting problem. Their approach is based on integrating the model structures of ensemble constituents into a single model. This model still provides a process-based explanation of the observed system structure, while being more robust in terms of over-fitting observed data. The evaluation of overfitting is performed by a variant of the general cross-validation method, where samples of data are kept out of the training set and are used to estimate the model error. While this method provides estimates of model error on unseen data, these estimates are not related to the predictive performance of the model, i.e., its ability to predict future system behaviour beyond the time-period captured in training data.

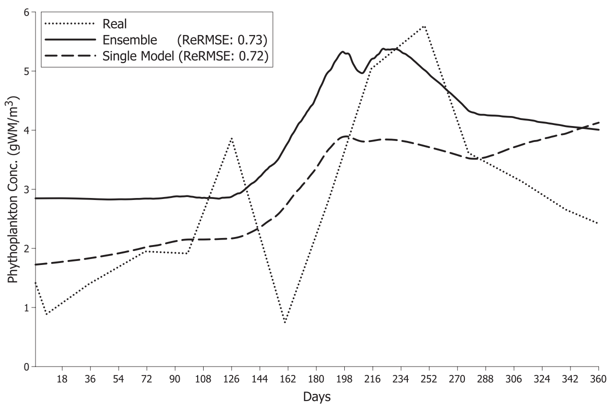
The studies of Whigham and Recknagel (2001) and Cao et al. (2008) are also related to our work, as they use differential equations to model the dynamics of lake ecosystems. However, they start from a modelling assumption that includes a fixed structure of model equations and employ genetic algorithms to estimate the values of the model parameters. While Whigham and Recknagel



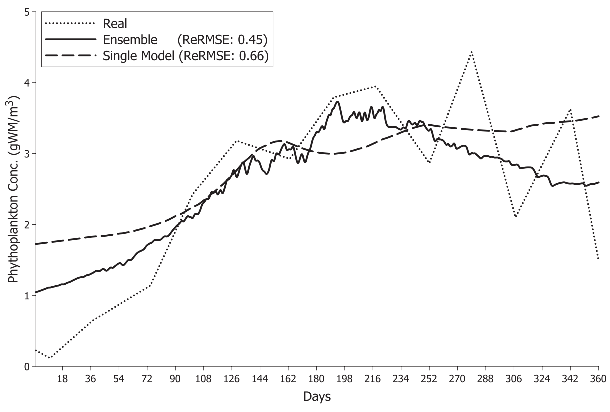
(a) Test Bled 4



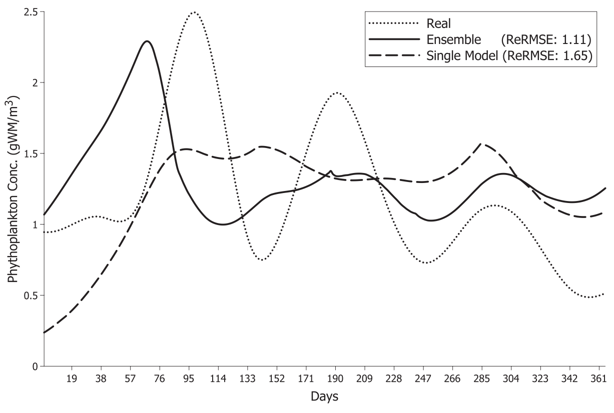
(b) Train Bled 4



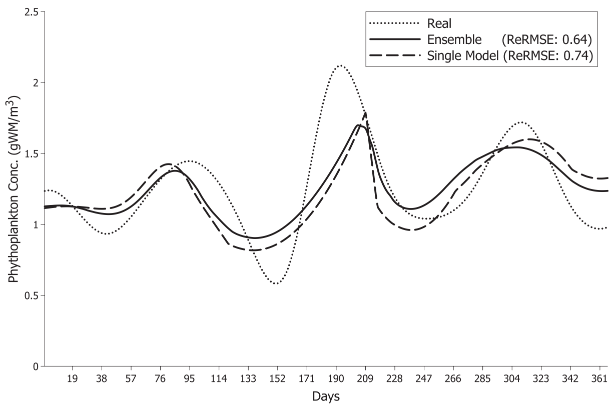
(c) Test Kasumigaura 5



(d) Train Kasumgaura 5



(e) Test Zurich 5



(f) Train Zurich 5

Fig. 9. Simulations of single models and ensembles compared to the measured data, for three pairs of test and training data sets.

(2001) also consider the use of genetic programming to explore a number of different model structures, the obtained equations are not cast in the form of process-based models and therefore do not provide insight into the processes and entities that govern the dynamics of the observed systems.

Muttil and Chau (2006) also use genetic programming and artificial neural networks to model algal blooms in coastal marine ecosystems. While their aim is similar to ours, i.e., to obtain predictive models of algal biomass dynamics, they focus exclusively on black-box models of population dynamics. The models they consider are based on equations and neural networks and fail to provide insight into the structure of the observed system. They also consider a different time scale for predicting the future system behaviour: While we focus in our experiments on a whole-year prediction the population dynamics, Muttil and Chau (2006) use their models for making one-week-ahead predictions of algal blooms.

The ensemble method proposed in this paper aims at improving the generalization power of process-based models, in terms of achieving predictive performance gain over the state-of-the-art process-based modelling approaches. However, when learning ensembles of process-based models, there is a trade-off between two conflicting requirements: predictive performance and interpretability. The increase of predictive accuracy comes at the cost of losing the interpretability of the learned ensemble model. Moreover, Breiman (1996a) states that bagging can improve the predictive performance when the ensemble is comprised of unstable base-models, such as decision trees, whose predictions sufficiently vary with small variations in the training set. In this case, high diversity of the ensemble constituents can be easily achieved. Our empirical evaluation shows that the ensemble constituents have only modest diversity: This may limit the potential for performance gain, even though the diversity is only weakly correlated with predictive performance.

7. Conclusion

7.1. Summary

In this paper, we address the task of learning ensembles of process-based models of dynamic systems and develop a methodology to solve it. Note that the task of learning ensembles of process-based models is a novel task, and has not been considered so far. For this purpose, we extend the state-of-the-art approaches to process-based modelling: We take the notion of ensembles – a collection of base-models, whose predictions are combined to improve the collective performance. In traditional machine learning, this has proved to be an effective method for gaining predictive power.

More specifically, we propose a methodology that adapts the key design principles from learning ensembles for classical machine learning tasks to tasks of modelling dynamic systems. Our approach constructs homogeneous ensembles of process-based models, using bagging as an underlying ensemble method. We identify the main components of the method for learning an ensemble of PBMs (a technique for learning a set of candidate base-models, a technique for selecting the base-models, and a combining scheme specifying how the base-model predictions are combined) and the related design choices.

We conduct an extensive experimental evaluation to identify the appropriate design choices for the proposed ensemble learning method and to test its utility for modelling dynamic systems. We analyze the improvement of performance obtained by ensembles relative to single models on 15 different data sets. Moreover, we investigate the relation between the diversity of the ensemble constituents and the performance improvement. Finally, we visually compare the simulations of ensembles with the simulations of single models in both descriptive and predictive scenarios.

We conduct the empirical evaluation on the task of modelling population dynamics in aquatic ecosystems. The case studies considered concern modelling phytoplankton growth, a complex non-linear dynamic process, in three different aquatic ecosystem domains. These include: Lake Bled in Slovenia, Lake Kasumigaura in Japan and Lake Zurich in Switzerland.

The results of the empirical evaluation confirm our central hypothesis. For predictive modelling tasks, ensembles of process-based models perform better than a single model. More precisely, ensembles with a relatively low number of constituents (10–25), chosen on a separate validation data set and combined by averaging, outperform the single model. Moreover, the diversity of the constituents in the ensemble is positively (weakly) correlated with the performance improvement. Finally, after visual inspection of the simulations, we found that the ensembles are applicable to both predictive and descriptive modelling tasks.

7.2. Future work

We have identified a number of limitations of our approach that can be addressed in further work. First, the diversity of the ensemble constituents is modest. One reason for this might be the fact that we use a library with a limited number of modelling alternatives. An extended library of domain-specific knowledge should be used in future experiments in order to reach higher diversity and further study the relation between the diversity and predictive performance in ensembles of process-based models.

Future work can also include the development of alternative methods for data and knowledge sampling that would lead to higher diversity. These include sampling the data variables (in addition to sampling the data instances considered in this paper) and sampling the alternative modelling choices included in the library of domain-specific modelling knowledge. While we limited our focus in this paper on adapting the method of bagging, further work should adapt other ensemble methods to the task of learning process-based models: Methods to be adapted include *boosting* (Drucker, 1997; Freund, 1999; Schapire, 2003) and *random subspaces* (Ho, 1998).

Finally, the experiments performed in this paper were limited to modelling population dynamics in lake ecosystems from historical data. Future experiments can be based on more recent data of the same ecosystems. Also, future work should confirm the results presented in this paper by learning ensembles of process-based models of population dynamics in other aquatic environments, such as marine ecosystems or water-treatments plants (Škerjanec et al., 2014). Other application domains (such as systems neuroscience and systems biology) should also be considered.

Acknowledgements

We thank Panče Panov and Dragi Kocev for the helpful discussions and comments on the work presented in this paper. Thanks to Nataša Atanasova for providing the extensive library for modelling aquatic ecosystems and the data sets for Lake Bled and Lake Kasumigaura. We also would like to thank Anne Dietzel for providing data on Lake Zurich. Finally, we acknowledge the financial support of the European Commission through the project SUMO – Supermodelling by combining imperfect models (grant number ICT-2009-266722) and MAESTRA – Learning from Massive, Incompletely annotated, and Structured Data (grant number ICT-2013-612944).

References

Čerepnalkoski, D., Taškova, K., Todorovski, L., Atanasova, N., Džeroski, S., 2012. The influence of parameter fitting methods on model structure selection in automated modeling of aquatic ecosystems. Ecol. Model. 245, 136–165.
Škerjanec, M., Atanasova, N., Čerepnalkoski, D., Džeroski, S., Kompare, B., 2014. Development of a knowledge library for automated watershed modeling. Environ. Model. Softw. 54, 60–72.
Atanasova, N., Recknagel, F., Todorovski, L., Džeroski, S., Kompare, B., 2006a. Computational assemblage of Ordinary Differential Equations for Chlorophyll-a using a lake process equation library and measured data of Lake Kasumigaura. In: Recknagel, F. (Ed.), Ecological Informatics. Springer, pp. 409–427.

Atanasova, N., Todorovski, L., Džeroski, S., Kompare, B., 2006b. Constructing a library of domain knowledge for automated modelling of aquatic ecosystems. Ecol. Model. 194 (1–3), 14–36.

Atanasova, N., Todorovski, L., Džeroski, S., Remec, R., Recknagel, F., Kompare, B., 2006c. Automated modelling of a food web in Lake Bled using measured data and a library of domain knowledge. Ecol. Model. 194 (1–3), 37–48.

Breiman, L., 1984. Classification and Regression Trees. Chapman & Hall, London, UK.

Breiman, L., 1996a. Bagging predictors. Mach. Learn. 24 (2), 123–140.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.

Bridewell, W., Asadi, N.B., Langley, P., Todorovski, L.,2005. Reducing overfitting in process model induction. In: Proceedings of the 22nd International Conference on Machine Learning (ICML '05). ACM, pp. 81–88.

Bridewell, W., Langley, P.W., Todorovski, L., Džeroski, S., 2008. Inductive process modeling. Mach. Learn. 71, 1–32.

Cao, H., Recknagel, F., Cetin, L., Zhang, B., 2008. Process-based simulation library SALMO-OO for lake ecosystems. Part 2: Multi-objective parameter optimization by evolutionary algorithms. Ecol. Inform. 3 (2), 181–190.

Cohen, S.D., Hindmarsh, A.C., 1996. CVODE, a stiff/nonstiff ODE solver in C. Comput. Phys. 10 (March (2)), 138–143.

Crisci, C., Ghattas, B., Perera, G., 2012. A review of supervised machine learning algorithms and their applications to ecological data. Ecol. Model. 240, 113–122.

Džeroski, S., Todorovski, L.,1993. Discovering dynamics. In: Proceedings of Tenth International Conference on Machine Learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 97–103.

Džeroski, S., Todorovski, L., 2003. Learning population dynamics models from data and domain knowledge. Ecol. Model. 170, 129–140.

Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7 (December), 1–30.

Dietzel, A., Mieleitner, J., Kardaetz, S., Reichert, P., 2013. Effects of changes in the driving forces on water quality and plankton dynamics in three Swiss lakes – long-term simulations with BELAMO. Freshw. Biol. 58 (1), 10–35.

Drucker, H.,1997. Improving regressors using boosting techniques. In: Proceedings of the 14th International Conference on Machine Learning (ICML '97). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 107–115.

Durillo, J.J., Nebro, A.J., 2011. jMetal: A Java framework for multi-objective optimization. Adv. Eng. Softw. 42, 760–771.

Freud, Y., Schapire, R.E., 1999. A short introduction to boosting. J. Jpn. Soc. Artif. Intell. 14 (5), 771–780.

Freund, Y.,1999. An adaptive version of the boost by majority algorithm. In: Proceedings of the 12th Annual Conference on Computational Learning Theory (COLT '99). ACM, New York, NY, USA, pp. 102–113.

Friedman, M., 1940. A comparison of alternative tests of significance for the problem of m rankings. Ann. Math. Stat. 11 (1), 86–92.

Ho, T.K., Aug 1998. The random subspace method for constructing decision forests. IEEE Trans. Pattern Anal. Mach. Intell. 20 (8), 832–844.

Iman, R.L., Davenport, J.M., 1980. Approximations of the critical region of the Friedman statistic. Commun. Stat. – Theory Methods 9 (6), 571–595.

Knudby, A., Brenning, A., LeDrew, E., Feb. 2010. New approaches to modelling fish–habitat relationships. Ecol. Model. 221 (3), 503–511.

Kuncheva, L.I., Whitaker, C.J., 2003. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Mach. Learn. 51 (May (2)), 181–207.

Langley, P.W., Simon, H.A., Bradshaw, G., Zytkow, J.M., 1987. Scientific Discovery: Computational Explorations of the Creative Processes. The MIT Press, Cambridge, MA, USA.

Luenberger, D., 1979. Introduction to Dynamic Systems: Theory, Models, and Applications. Wiley, NJ, USA.

Maclin, R., Opitz, D., 1999. Popular ensemble methods: An empirical study. J. Artif. Intell. Res. 11, 169–198.

Muttil, N., Chau, K.W., 2006. Neural network and genetic programming for modelling coastal algal blooms. Int. J. Environ. Pollut. 28 (3–4), 223–238.

Nemenyi, P.B., 1963. Distribution-free Multiple Comparisons. Princeton University, Princeton, NJ, USA (Ph.D. thesis).

Okun, O., Valentini, G., Ré, M. (Eds.), 2011. Ensembles in Machine Learning Applications. Vol. 373 of Studies in Computational Intelligence. Springer, Berlin.

Rokach, L., Feb. 2010. Ensemble-based classifiers. Artif. Intell. Rev. 33 (1–2), 1–39.

Schapire, R.E., 2003. The boosting approach to machine learning: an overview. In: Denison, D., Hansen, M., Holmes, C., Mallick, B., Yu, B. (Eds.), Nonlinear Estimation and Classification. Vol. 171 of Lecture Notes in Statistics. Springer, New York, pp. 149–171.

Storn, R., Price, K., 1997. Differential Evolution - A simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11 (4), 341–359.

Taškova, K., Korošec, P., Šilc, J., Todorovski, L., Džeroski, S., 2011. Parameter estimation with bio-inspired meta-heuristic optimization: modeling the dynamics of endocytosis. BMC Syst. Biol. 5 (1), 159.

Taškova, K., Šilc, J., Atanasova, N., Džeroski, S., 2012. Parameter estimation in a nonlinear dynamic model of an aquatic ecosystem with meta-heuristic optimization. Ecol. Model. 226, 36–61.

Todorovski, L., Džeroski, S., 2007. Integrating domain knowledge in equation discovery. In: Džeroski, S., Todorovski, L. (Eds.), Computational Discovery of Scientific Knowledge. Vol. 4660 of Lecture Notes in Computer Science. Springer, Berlin, pp. 69–97.

Todorovski, L., Bridewell, W., Shiran, O., Langley, P.W.,2005. Inducing hierarchical process models in dynamic domains. In: Proceedings of the 20th National Conference on Artificial Intelligence. AAAI Press, Pittsburgh, USA, pp. 892–897.

Whigham, P., Recknagel, F., 2001. Predicting Chlorophyll-a in freshwater lakes by hybridising process-based models and genetic algorithms. Ecol. Model. 146 (1–3), 243–251.

Wolpert, D.H., 1992. Stacked generalization. Neural Netw. 5, 241–259.

Zhou, Z.-H., Wu, J., Tang, W., 2002. Ensembling neural networks: Many could be better than all. Artif. Intell. 137 (1–2), 239–263.

Expert Systems With Applications 42 (2015) 8484–8496

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Predicting long-term population dynamics with bagging and boosting of process-based models

Nikola Simidjievski[a,b,*], Ljupčo Todorovski[c], Sašo Džeroski[a,b]

[a] *Department of Knowledge Technologies, Jožef Stefan Institute, Jamova cesta 39, Ljubljana 1000, Slovenia*
[b] *Jožef Stefan International Postgraduate School, Jamova cesta 39, Ljubljana 1000, Slovenia*
[c] *Faculty of Administration, University of Ljubljana, Gosarjeva ulica 5, Ljubljana 1000, Slovenia*

A R T I C L E I N F O

Keywords:
Ensembles
Process-based modeling
Bagging
Boosting
Machine learning
Predictive modeling
Population dynamics

A B S T R A C T

Process-based modeling is an approach to learning understandable, explanatory models of dynamic systems from domain knowledge and data. Although their utility has been proven on many tasks of modeling dynamic systems in various domains, their ability to accurately predict the future behavior of an observed system is limited. To address this limitation, we propose the use of a standard approach to improving the predictive performance of machine learning methods, i.e., the approach of learning ensemble models. Previous work on ensembles of process-based models has been limited to proof-of-principle experiments with a single ensemble method (bagging) and in the limited perspective of explaining the currently observed system behavior v.s. predicting future system behavior. In this paper, we design a general methodology for adapting ensemble methods to the context of process-based modeling. Using the methodology, we implement the two approaches bagging and boosting of process-based models. We perform an empirical evaluation of the implemented methods on three real-world modeling problems from the domain of population dynamics in aquatic ecosystems. The results of the empirical evaluation show that ensembles of process-based models can lead to long-term predictions of the population dynamics that are more accurate than the ones obtained with a single process-based model.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

Mathematical models are employed to provide an understanding of the laws that govern the behavior of dynamic systems. More specifically, such models are being utilized to recreate or simulate the behavior of dynamic systems under various conditions. This paper addresses the task of automated modeling of dynamic systems from time-series data and domain-specific modeling knowledge. The result of which is a process-based model that both explains the structure of the modeled system and allows for simulation of its behavior (Todorovski & Džeroski, 2007). For simulation, process-based models are transformed to systems of ordinary differential equations (ODEs), a widely accepted formalism for modeling dynamic systems. ODEs allow for long-term simulation of the system behavior given only its state at the initial time point and the time series corresponding to the input control variables. Models that allow for accurate long-term prediction of system behavior are typically hand-crafted by engineers and experts in the domain at hand. The process-based modeling approach allows for automated learning of such models in different domains (Atanasova, Recknagel, Todorovski, Džeroski, & Kompare, 2006a; Atanasova et al., 2006c; Taškova, Šilc, Atanasova, & Džeroski, 2012).

However, existing approaches to process-based modeling mostly focus on building descriptive models that explain the observed behavior of the system, however do not generalize well enough to predict future system behavior (Simidjievski, Todorovski, & Džeroski, 2015). Improving the predictive performance of process-based models is still a challenge. In this paper, we address this challenge by proposing ensembles of such models. This is a standard approach for improving the predictive performance of models in machine learning Dietterich (2000). An ensemble is a set of models, referred to as base models or ensemble constituents; its prediction is a combination of the predictions obtained with the individual ensemble constituents. They are usually employed in the context of supervised learning tasks of classification (Smith et al., 2015) and regression (King, Abrahams, & Ragsdale, 2014; Tay, Chui, Ong, & Ng, 2013) to address the problems of over-fitting, high dimensionality, or missing features in the training data, resulting in predictive performance gain as compared to that of a single model. While regression ensembles can be also used as models for time-series forecasting (Ma, Dai, & Liu, 2015), they have

[*] Corresponding author at: Department of Knowledge Technologies, Jožef Stefan Institute, Jamova cesta 39, Ljubljana 1000, Slovenia. Tel.: +386 1 477 3635.
E-mail addresses: nikola.simidjievski@ijs.si (N. Simidjievski), ljupco.todorovski@fu.uni-lj.si (L. Todorovski), saso.dzeroski@ijs.si (S. Džeroski).

http://dx.doi.org/10.1016/j.eswa.2015.07.004
0957-4174/© 2015 Elsevier Ltd. All rights reserved.

so far been used to make short-term predictions, i.e., predict only the values of the system variables in immediate future and not the long-term system behavior.

The main motivation for the work presented in this paper is to improve the accuracy of long-term predictions of process-based models. To this end, our specific objective is to design and implement methods for learning ensembles of process-based models from data and knowledge. We conjecture that ensembles learned using the implemented methods will outperform single process-based models in terms of their accuracy/error of long-term prediction of system behavior. To test the validity of this conjecture, we perform an extensive evaluation of the implemented methods on the task of modeling and predicting population dynamics in aquatic ecosystems. The empirical evaluation allows us to decide between the ensemble methods of bagging (Breiman, 1996a) and boosting (Drucker, 1997; Freund, 1999) as well as design alternatives considered within this methods.

The remainder of the paper is organized as follows. In Section 2, we put our work in the context of related work on process-based modeling and ensemble learning. Section 3 provides an introduction to the process-based modeling paradigm and its specific implementation PRoBMoT (Čerepnalkoski, Taškova, Todorovski, Atanasova, & Džeroski, 2012) through an example modeling task from the domain of population dynamics in aquatic ecosystems. Section 4 presents the two methods we propose and their implementation as an extension to PRoBMoT for learning ensembles of process-based models. In Section 5, we present the experimental setup of the empirical framework for evaluating the developed methods on three tasks of modeling population dynamics in aquatic ecosystems. More specifically, we test the utility of ensembles of process-based models in the context of modeling three ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan, and Lake Zurich in Switzerland. Section 6 presents the results of the empirical evaluation and discusses them in the context of related research. Finally, Section 7 concludes this paper and suggests directions for further work.

2. Related work

The work presented in this paper extends the state-of-the-art of process-based modeling (Bridewell, Langley, Todorovski, & Džeroski, 2008; Todorovski & Džeroski, 2007). More specifically, it relates to previous work on process-based modeling that has proven successful for building descriptive models of population dynamics in a number of real-world aquatic ecosystems (Atanasova et al., 2006a, 2006c; Džeroski & Todorovski, 2003). These studies focus on descriptive models used to explain the observed behavior of the system at hand and not (so far) employed for predicting future (unobserved) system behavior. Simidjievski et al. (2015) present a proof-of-principle experiment and show that bagging process-based models can improve their predictive performance. However, this experiment is of limited scope since it does not consider alternative ensemble methods, such as boosting, which is considered here. The study presented there is purely empirical and does not tackle the methodological issues of learning ensembles that are being addressed in this paper.

The task of learning ensembles of process-based models has been also addressed by Bridewell, Asadi, Langley, and Todorovski (2005), who aim at integrating the ensemble constituents into a single (meta-level) process-based model that includes the structure fragments most frequently present in the constituents. The results show that the resulting meta-level model is more robust in terms of over-fitting to the observed data. Note however, that in the evaluation their proposed approach, the authors estimate the out-of-sample error by taking random sub-samples of the observed (training) time-series data. The ability of the meta-level model to predict system behavior outside the observed time span of the training data (i.e., future system behavior), which we focus on in this paper, has not been considered, far less evaluated.

On the other hand, our work is related to the long tradition of applying methods for learning ensembles to various predictive modeling tasks in different scientific and engineering domains. The ensemble methods that tackle the problem of time-series forecasting are most closely related. (Ma et al., 2015) present a method for pruning the set of ensemble constituents, in particular support-vector regression models, for the purpose of optimizing the size of an ensemble and its predictive performance. They apply the developed method to four tasks of forecasting financial time-series (stock indices) and show that the approach can reduce ensemble size while retaining reasonable levels of predictive accuracy. Similarly, Kourentzes, Barrow, and Crone (2014) also aim at short-term forecasting of financial time series using ensembles of neural networks. Their main contribution is the identification of the most suitable operator for aggregating the predictions of individual ensemble constituents. Note that all the forecasting tasks, considered in these two papers, are short-term since they aim at forecasting the next-time-point values of stock indices or retail prices. In contrast, our process-based models aim at long-term (typically one year) prediction that concern periods with potentially indefinite ranges of time points.

Finally, process-based modeling is a term frequently used in the realm of system analytics and modeling business processes. Recently, there has been interest in automatic building of enactment plans related to the same declarative specification of a given business process (Jiménez, Barba, del Valle, & Weber, 2013). Business process modeling formalisms follow the standard specification languages, such as BPMN (White, 2004), designed for the specific purpose of specifying declarative models of business processes that can be used for understanding, simulating and optimizing business processes. However, the business process models are quite different from the process-based models considered here that provide means for a mathematical formalization of the quantitative change of the observed system state through time. Our process-based models are thus based on an entirely different formalism that we will introduce in the next section.

3. Process-based modeling

Complex models are derived with the express purpose to recreate the observed behavior or simulate the subsequent states of a dynamic system under various conditions. Scientist and engineers often relate such models to the processes that govern the dynamics of the modeled system, and to the entities involved. These relations are commonly construed with equation-based specifications of the dynamics, and compiled into a set of ordinary differential and/or algebraic equations. The set of equations describes the change of the system's state over time and therefore is used to simulate past, present and future behavior of the system at hand. However, while equations offer a quantitative way of expressing models, they lack the ability to qualitatively express the structure of the modeled system in terms of interacting entities and processes.

The approach of process-based modeling of dynamic systems aims at constructing models which contain a high-level explanatory structure and a low-level mathematical formulation which allowing for making predictions. Process-based models integrate domain-specific modeling knowledge and data into explanatory models of the observed systems. A process-based model consists of two basic types of elements: entities and processes. Entities represent the state of the system. They incorporate the input variables (forcing terms of the system), state variables (the internal state of the system) and the constants related to the components of the modeled system. The entities are involved in complex interactions represented by the processes. The processes include specific details of how the entities interact in terms of equations and sub-processes.

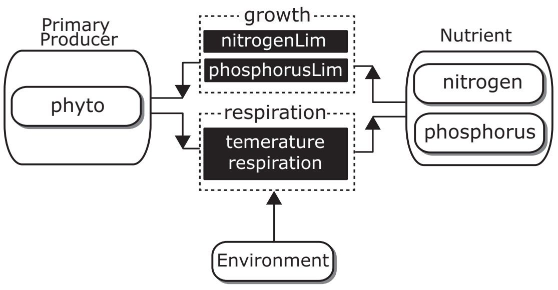


Fig. 1. Graphical representation of the relations (arrows and black boxes) between the entities (oval transparent boxes) in a simple lake ecosystem.

The task of process-based modeling, or learning process-based models from knowledge and data, can be specified, in terms of its inputs and outputs, as follows:

Input

– Measured data of the variables in the observed system.
– Domain-specific modeling knowledge.

Output

– Process-based model of the observed system.

The measured values of the observed variables are continuous, contiguous and may be non-uniformly distributed. We are interested in models that can predict the behavior of the system, i.e., how its state changes over time. Since, the state is represented by continuous variables, the task at hand resembles the task of regression.

Regression models, applied in the context of time-series data, are typically used for short-term prediction of the state at the next time point, based on the observed values of the current and previous states. In contrast, differential equations provide long-term predictions over many following time points, based only on the initial values of the target variables; no observations are necessary at the intermediate time points. Note also, that the use of modeling knowledge is an advantage of the process-based modeling approach over regression approaches since it improves model interpretability. The process-based models upgrade the purely empirical approaches, and strive at explaining how and why the dynamic system behaves under various conditions as opposed to just explaining/predicting how the measurements vary.

In the continuation of this section we are going to explain in more detail the process-based modeling paradigm through the prism of process-based models and the process of their learning from knowledge and data. To properly asses the relevant details of the paradigm, we are going to illustrate its use on a simple example of modeling population dynamics in an aquatic ecosystems.

3.1. Process-based models and modeling knowledge

Models of aquatic ecosystems are required for better understanding, prediction and management of such systems (Jørgensen & Bendoricchio, 2001). These models target the relations between entities, i.e., nutrients, primary producers, animals and environmental changes, that typically occur in aquatic ecosystems (Luenberger, 1979). Fig. 1 depicts a cyclic relationship involving a primary producer (phytoplankton, abbreviated as phyto) that grows by feeding on nutrients (nitrogen and phosphorous), the concentrations of which are influenced by the environment and the process of phytoplankton respiration (Atanasova, Todorovski, Džeroski, & Kompare, 2006b).

In order to model such a system using the process-based paradigm, we first need to formalize the modeling knowledge. Process-based modeling allows for a high-level representation of knowledge, cataloged in a domain-specific library of entity and process templates. The templates embody general properties of the interactions that govern the dynamics in the domain at hand and serve as recipes for establishing specific entities and processes observed in a given system.

Table 1 depicts an example of a simple library for modeling population dynamics in aquatic ecosystems. The first four declarations correspond to template entities organized in a hierarchy. The template entities of *EcosystemEntity* and *Environment* are at the top of the hierarchy; the first corresponds to the entities of the aquatic ecosystem, while the second to its environment. Down the hierarchy, the *EcosystemEntity* is then specialized into the two template entities of *PrimaryProducer* and *Nutrient*.

Each entity template may include constant and variable properties, which are inherited down the hierarchy. The variable properties (denoted `vars`) are those which change over time: the *EcosystemEntity* has as variable property its current concentration (denoted *conc*). Similarly, the *Environment* entity includes the *temperature* variable. The constant properties that do not change are denoted with `consts`, e.g., the template entity *PrimaryProducer* has the constant property *maxGrowthRate*. Finally, note that each variable property can have an aggregation function (denoted `aggregation`) that specifies how multiple influences on the variable (originating from different processes) are combined: the influences on the *conc* variable of *EcosystemEntity* are summed up, while the influences on the *growthRate* variable are multiplied.

The template processes are also organized in a hierarchy and specify which entities can interact and how these interactions govern the dynamics of entity variables. Highest in the hierarchy of aquatic ecosystem processes are the templates of GROWTH, RESPIRATION and GROWTHRATE. The RESPIRATION template specifies two differential equations that model the influence of the respiration process on the concentrations of the primary producer and the nutrients involved. Similarly, the GROWTH template encodes the influences of growth on the same concentrations. Additionally, GROWTH involves a subprocess GROWTHRATE, which implies that a GROWTHRATE must be specified for each nutrient involved in the process of growth. The hierarchy specifies two instances of the template process GROWHRATE, MONODGROWTHRATE and EXPSATURATEDGROWTHRATE, that correspond to two alternative models of growth limitation due to limited nutrient supply. Note therefore, the hierarchical structure of the process templates allows for the specification of modeling alternatives for an observed interaction between entities.

Once we have a library of process and entity templates, we can formulate a process-based model as a set of instances of the templates in the library. Table 2 presents a process-based model of the system depicted in Fig. 1: note the one-to-one correspondence between entities and process depicted in the system graphical presentation and the process-based model. Each entity and process instance incorporate the variables and the constants related to the corresponding template. For each variable property, its role in the observed system is specified. Exogenous variables represent input system variables that are not the subject of modeling (e.g., the environment temperature), while endogenous variable represent system state variables that are subject to modeling (e.g., the concentration of phytoplankton). For each endogenous variable, we have to provide its value at the initial time point. Finally, for the constant properties of the entities and processes, we have to specify their values.

Considering the mathematical formulation of the processes embodied in the library, we can compile this high-level representation of the interactions in the system into a system of algebraic and differential equations adequate for simulation. Table 3 provides the quantitative formulation of the process-based model presented

Table 1
Template entities and processes for modeling population dynamics in aquatic ecosystems. Here td(x) denotes the time derivative of x.

```
template entity EcosystemEntity{
   vars:conc{aggregation:sum;}}
template entity PrimaryProducer:EcosystemEntity{
   vars:growthRate{aggregation:product};
   consts:maxGrowthRate; }
template entity Nutrient:EcosystemEntity{consts:alpha;}
template entity Environment{vars:temperature;}
template process Growth(pp:PrimaryProducer,ns:Nutrients){
   processes:GrowthRate(pp,ns);
   equations:
     td(pp.conc) = pp.maxGrowthRate * pp.growthRate * pp.conc,
     td(ns.conc) = −n.alpha * pp.maxGrowthRate * pp.growthRate * pp.conc; }
template process Respiration(
pp:PrimaryProducer,ns:Nutrients,env:Environment){
   consts:respRate,refTemp,minTemp;
   equations:
     td(pp.conc) = −respRate * pp.conc * pp.conc
              *(env.temperature-minTemp)/(refTemp-minTemp),
     td(ns.conc) = respRate * pp.conc * pp.conc
              *(env.temperature-minTemp)/(refTemp-minTemp); }
template process GrowthRate(pp:PrimaryProducer,n:Nutrient){}
template process MonodGrowthRate:GrowthRate{
   consts:halfSaturation;
   equations:
     pp.growthRate = n.conc/(n.conc + halfSaturation);}
template process ExpSaturatedGrowthRate:GrowthRate{
   consts:saturationRate;
   equations:
     pp.growthRate = 1 − exp( − saturationRate * n.conc);}
```

Table 2
A process-based model of phytoplankton dynamics in the simple a lake ecosystem from Fig. 1, based on the template entities and processes from the library in Table 1.

```
//Entities
entity phyto:PrimaryProducer{
   vars:conc{role:endogenous; initial: 1.665;};
   consts:maxGrowthRate= 0.88;}
entity phos:Nutrient{
   vars:conc{role:exogenous;};}
entity nitro:Nutrient{
   vars:conc{role:exogenous;};}
entity env:Environment{
   vars:temp{role:exogenous;};}
//Processes
process growth(phyto, [phos, nitro]):Growth
   {processes:growthRate;}
process nitrogenLim(phyto,nitro):ExpSaturatedGrowthRate
   {consts:saturationRate=14.9;}
process phosophorousLim(phyto,phos):ExpSaturatedGrowthRate
   {consts:saturationRate=8.08;}
process respiration(phyto, [phos,nitro],env):Respiration
   {consts:respRate=0.036,minTemp=0.542,refTemp=17.4;}
```

Table 3
Ordinary differential equations obtained from the process-based model of phytoplankton dynamics presented in Table 2. Here td(x) denotes the time derivative of x.

$$\text{phyto.growthRate} = [1 - \exp(-8.08 * \text{phos.conc})] * [1 - \exp(-14.9 * \text{nitro.conc})]$$
$$\text{td}(\text{phyto.conc}) = 0.88 * \text{phyto.conc} * \text{phyto.growthRate} - \text{phyto.conc}^2 * 0.036 * \frac{\text{env.temp} - 0.542}{\text{env.temp} - 17.4}$$
$$\text{phyto.conc}(t_0) = 1.665$$

above. Since we are modeling just the concentration of phytoplankton (which is denoted as endogenous in Table 2) the system of equations consists of a single differential equation. This equation can be then simulated, thus generating a trajectory and utilizing it for further analysis. Fig. 2 shows the simulation of phytoplankton concentration obtained by using the process-based model. The *data* trajectory (represented by a dashed line) represents real measurements that can be used for a visual assessment of the process-based model performance.

In summary, process-based models have several characteristics which make them very efficient for tasks of modeling dynamic systems. First, they provide a conceptual representation of the structure of the modeled system, depicting the high-level relations (processes) between the system components (entities). Second, they allow for the high-level process-based representation to be translated into a low-level mathematical formalism depicted as a set of differential and/or algebraic equations, facilitating simulation of the systems behavior. Finally, the library of domain-specific knowledge allows for

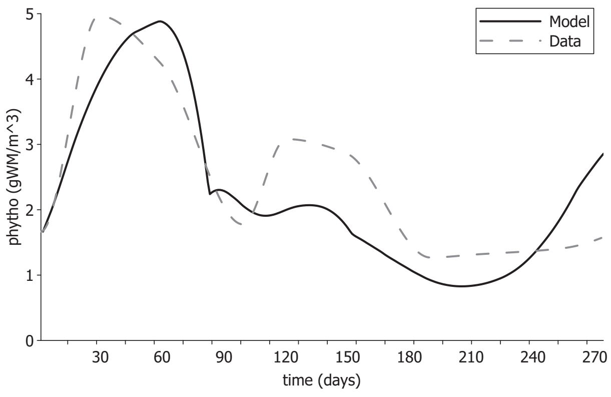


Fig. 2. Simulation of phytoplankton concentration dynamics (solid line) as modeled with the process-based model from Table 2 and its comparison to observed phytoplankton concentration (dashed line).

Algorithm 1 Outline of the generic algorithm for learning process-based models from knowledge and data.

```
1: function PBM($library, data, incompleteModel$)
2:     $components \leftarrow$ INSTANTIATE($library, incompleteModel$)
3:     for each ($structure \in$ ENUMERATE($components,$
       $incompleteModel$)
   do
4:         $modelEq \leftarrow$ COMPILETOEQUATION($strucutre$)
5:         $(model, error) \leftarrow$ PARAMETERESTIMATION($modelEq, data$)
6:         $modelList \leftarrow modelList \bigcup (model, error)$
7:     end for
8:     return SORT($modelList, error$)
9: end function
```

instantiation of a number of different building blocks for generating process-based models (Bridewell et al., 2008), which is particularly relevant for algorithms tackling the task of automated learning of process-based models from data.

3.2. Learning process-based models

Given the library of model fragments (template entities and processes), we can now formulate the task of learning process-based models from knowledge and data as a search task. Namely, given the specific entities in the observed system at hand, one can instantiate the template processes from the library into a set of specific processes that can be considered for inclusion in the model of the observed system. In turn, based on this set of specific model components, we can specify the search space of combinations thereof. Some of the combinations can be rejected as implausible, due to further modeling assumptions made by the user, such as the presence or absence of certain processes in the model.

The process-based modeling (PBM) algorithm, outlined in Algorithm 1, first takes as input *library* of domain-specific modeling knowledge, followed by *data* in form of time-series measurements of the observed dynamic system. The last input to the algorithm is an *incompleteModel* representing the modeling assumptions made by the modeler. First, the algorithm assembles all theoretically plausible model components by binding the entities of the observed system to the template processes from the library. Next, based on the incomplete model (taking into account the assumptions), the algorithm enumerates all the plausible candidate model structures. Each of these high-level structures is then compiled into a system of equations eligible for simulation. Before simulation, however, a parameter estimation task is being solved for the model structure at hand, to obtain values of the model parameters that best fit the observed data. After estimating the parameters for all candidate model structures, the algorithm outputs a sorted list of process-based models according to their error on training data, i.e., the discrepancy between the model simulation and the observed system behavior.

Most process-based modeling algorithms perform exhaustive search through a constrained space of candidate process-based structures, limiting the number of processes in the model (Bridewell et al., 2008). The more advanced approaches, such as Lagramge2.0 (Todorovski & Džeroski, 2007), combine the library of knowledge and the constraints into a grammar for enumerating plausible model structures. HIPM (Todorovski, Bridewell, Shiran, & Langley, 2005) allows for more sophisticated hierarchical constraints on the legal process combinations and tackles enumeration of model structures as a combinatorial problem. Last, ProBMoT (Čerepnalkoski et al., 2012) is a software platform for complete modeling, parameter estimation, and simulation of process-based models. It extends HIPM with explicit constraints (assumptions) for a particular domain at hand and employs a variety of meta-heuristic optimization methods. In this study, we use ProBMoT[1] as the base learning algorithm for learning constituents of ensembles of process-based models.

4. Ensembles of process-based models

Ensembles are commonly used for machine learning tasks, such as classification and regression (Maclin & Opitz, 1999; Rokach, 2010). However, a detailed layout of a methodology for learning ensembles of process-based models for predictive tasks has not been considered so far. Here we take note of the similarity between the tasks of process-based modeling and time-series regression, and we apply the idea of learning ensembles of regression models in the context of models of dynamic systems, that is, developing appropriate methods for learning ensembles of process-based models.

An ensemble is a set of models (referred to as base models or ensemble constituents) that is expected to have improved predictive performance compared to a single model. The idea behind ensembles is to maximize the overall predictive power by combing the predictions of the individual base models. The simplest form of an ensemble is a black-box. For a given "bag" of individual models, the resulting output is a combination of the individual predictions. For this reason, we first explore how an ensemble of process-based models can be simulated and how the resulting prediction can be interpreted. Next, we "open" the black-box, and investigate the different methods for learning the constituents of an ensemble, where we introduce a novel approach of learning ensembles of process-based models.

4.1. Simulating ensembles of PBMs

In order to simulate an ensemble, each base model needs to be simulated. The resulting ensemble output is a combination of the predictions of all individual base models. For obtaining a prediction for ensembles of process-based models, we use average, weighted average and weighted median as combining schemes, commonly used for regression tasks (Drucker, 1997).

In all cases, the real-valued predictions of the constituent models are combined per time-point, for each time-point separately. In the case of average, all base models participate in the resulting simulation equivalently. For weighted average and weighted median schemes, a confidence β is calculated for each of the base models based on their performance error. The base models with higher confidence will contribute more in the resulting ensemble simulation. The procedure for simulating an ensemble of PBMs is depicted in Algorithm 2.

The SIMULATEENSEMBLE() procedure takes as input: a set of process-based models denoted with *ensemble*, the library of domain

[1] Available at http://probmot.ijs.si.

Algorithm 2 Simulating ensembles of process-based models.

```
1:  function SIMULATEENSEMBLE($ensemble, lib, D, scheme$)  returns $\hat{y}_e$
2:      $Simulations \leftarrow \emptyset$          ▷ simulations from ensemble constituents
3:      $\hat{y}_e \leftarrow \emptyset$          ▷ $\hat{y}_e$: the resulting ensemble simulation
4:      for each $\{model, \beta\} \in ensemble$ do
5:          $\hat{y} \leftarrow$ SIMULATE($model, D$)
6:          if INRANGE($\hat{y}, lib$) then
7:              $Simulations \leftarrow Simulations \bigcup \{\hat{y}, \beta\}$
8:          else continue
9:          end if
10:     end for
11:     if $scheme = average$ then
12:         $\hat{y}_e \leftarrow$ AVERAGE($Simulations$)
13:     else if $scheme = weightedAverage$ then
14:         $\hat{y}_e \leftarrow$ WEIGHTEDAVERAGE($Simulations$)
15:     else
16:         $\hat{y}_e \leftarrow$ WEIGHTEDMEDIAN($Simulations$)
17:     end if
18: end function
```

knowledge *lib*, a data set *D* and a label *scheme* selecting the combination scheme used. The resulting prediction of the ensemble is a trajectory denoted with $\hat{y}_e$. First, each model from the set is simulated. The result of the prediction of a individual model for a data set *D* is a trajectory $\hat{y}$. Each model is accompanied with a confidence β, calculated based on the performance on a validation data set. We use this coefficient β in the weighted combining schemes. The pairs of trajectories and confidences $\{\hat{y}, \beta\}$ resulting from the simulation of the constituents in the ensemble is collected in the set *Simulations*.

In contrast to the task of obtaining an output from a regression model, where the resulting prediction is a single point for a given input, the task of predicting with process-based models is far more challenging. The simulation of a process-based model takes as input the initial values of the endogenous variables and the complete trajectories of the exogenous (forcing) variables. As output, it produces complete trajectories of the endogenous variables. In a predictive scenario, this can often lead to divergent trajectories and disastrous predictive misperformance. For this reason, we examine the simulated values of each prediction $\hat{y}$ whether they satisfy the range of constraints given in the library of background knowledge (line 6 in Algorithm 2). If a value from a prediction is outside the range specified in the library, the whole trajectory of that particular model is discarded, i.e., is not taken into account when calculating the resulting ensemble prediction. In this paper, we use this kind of dynamic ensemble pruning as a standard technique when selecting the ensemble constituents and simulating the ensemble prediction. Finally, the valid simulations (*Simulations*) along with the respective confidence β_v are combined in the resulting ensemble prediction.

4.2. Learning the constituents of an ensemble of PBMs

Theoretically, ensemble methods consist of two main components: a technique for learning a set of candidate base models, and a combining scheme specifying how the base model predictions are being aggregated into an ensemble prediction. Previously, we demonstrated how the base models are combined into an ensemble of process-based models. Here we focus on the methods for learning the ensemble constituents.

Based on how the constituents are learned, the ensembles can be homogeneous or heterogeneous. In homogeneous ensembles, the base models are learned with the same learning algorithm, but using different samples of the training data, where the sampling variants include: sampling of data instances as in bagging (Breiman, 1996a) and boosting (Freund, 1999); sampling of data features/attributes as in random subspaces (Ho, 1998); or both as in random forests (Breiman, 2001). On the other hand, in heterogeneous ensembles, the candidate base models are learned using different learning algorithms, possibly together with a combination function as in stacking (Wolpert, 1992).

Algorithm 3 Bagging process-based models.

```
1:  procedure BAGGING($lib, \{D_T, D_V\}, incompleteModel, k$)
2:  returns $Ensemble$
3:      $Ensemble \leftarrow \emptyset$          ▷ set of base models
4:      for $i = 1$ to $k$ do
5:          $D_S \leftarrow$ SAMPLE($D_T$)    ▷ randomly sample the training set $D_T$
6:          $modelList_i \leftarrow$ PROBMOT($lib, D_S, incompleteModel$)
7:          $bestModel_i \leftarrow$ RANK($modelList_i, D_V$)
8:          $\beta_i \leftarrow$CONFIDENCE($bestModel_i, D_V$)
9:          $Ensemble \leftarrow Ensemble \bigcup (bestModel_i, \beta_i)$
10:     end for
11: end procedure
12:
13: function CONFIDENCE($model, D$)  returns $\beta$
14:     let $\hat{y}$          ▷ simulated system variable $y$
15:     let $y$          ▷ measured system variable $y$
16:     $\hat{y} \leftarrow$SIMULATE($model, D$)
17:     $maxDisc \leftarrow \sup(|y_t - \hat{y}_t|)^2$     ▷ calculate max discrepancy between measurements $y$ and simulation $\hat{y}$, where $t = 0..N$ and $N$ is number of time-points in D
18:     $\bar{L} \leftarrow \sum_{t=0}^{N} \frac{|y_t - \hat{y}_t|^2}{maxDisc}$          ▷ calculate average loss
19:     $\beta \leftarrow \frac{\bar{L}}{1 - \bar{L}}$          ▷ calculate confidence
20: end function
```

Bagging (Bootstrap aggregation), developed by Breiman (1996a), is one of the first and simplest ensemble learning methods. This method uses bootstrap sampling with aggregation. First, randomly sampled data instances, with replacements, from the training set are used to obtain bootstrap replicates. Next, each base model is learned from a different bootstrap replicate.

Boosting refers to a general approach for obtaining an accurate prediction by combining several less accurate ones learned on a different distributions of the training data. The AdaBoost algorithm, proposed by Freund and Schapire (1997), is an implementation of the boosting approach for the task of classification. AdaBoost works iteratively; it uses different distributions of the training data for learning the base models at each iteration. Depending on the outcome of the past iteration this method decreases (for correct classification)/increases (for incorrect classification) the weights of every instance, thus changing the distribution for the subsequent iteration of training the model. In this way, the individual weak predictors focus on different instances, and their combination is more robust. In a similar fashion, the implementation of Drucker (1997) successfully tackles the problem of combining regressors using AdaBoost.

In the reminder of this section, we will describe the process of generating ensembles of PBMs, and identify the key design principles for extending the two methods outlined above (bagging and Adaboost) to learn ensembles of PBMs.

4.2.1. Bagging of PBMs

The procedure for bagging process-based models is presented in Algorithm 3. The procedure BAGGING() takes four inputs: a library of domain knowledge *lib*, data consisting of training data D_T and validation data D_V, an incomplete model *incompleteModel*, and an integer k denoting how many base models are to be generated. The output is a set of process-based models denoted with *Ensemble*. Using PROBMOT()(line 6), we learn a set candidate base models from different random samples D_S of the training data D_T . The PROBMOT() procedure follows the algorithm design principles of the process-based modeling paradigm, and resembles Algorithm 1 in terms of inputs, outputs and flow.

The notable difference from bagging in the context of regression is that in our case the data instances have a temporal ordering, that has to be retained in each data sample. To achieve this, we implement sampling by retaining the order of the instances and introducing a weight for each instance (time-point), and provide it as part of the data. The weight corresponds to the number of times the instance has been selected in the process of sampling with replacement (SAMPLE() procedure). Instances that have not been selected (the ones with weight 0) are simply omitted from the sample.

To take into account the weights when learning a model from the sample, we employ the weighted root mean squared error (WRMSE) implemented in ProBMoT:

$$WRMSE(m) = \sqrt{\frac{\sum_{t=0}^{N} \omega_t * (y_t - \hat{y}_t)^2}{\sum_{t=0}^{n} \omega_t}}, \tag{1}$$

where y_t and $\hat{y}_t$ correspond to the *measured* and *simulated* values (simulating the base model m) of the system variable y at time point t, N denotes the number of instances in the data sample, and ω_t denote the weight of the data instance at time point t.

The output of a modeling task, when using ProBMoT, is a list of process-based models, which is afterwards sorted according to their performance (line 7 in Algorithm 3). Depending on the input in the procedure, the ranking can be based on the performance on a separate validation data set D_V, or on the training sample (if $D_V == D_T$). The highest ranked model from each modeling task i (out of k) denoted as $bestModel_i$, becomes an ensemble constituent in the output *Ensemble*.

Each ensemble constituent is paired with its own confidence β. The CONDIFENCE() function takes 2 inputs: the highest ranked model returned by ProBMoT and a data set D. Fist the *model* is simulated on the data set D resulting in a trajectory $\hat{y}$. Based on the error at each time point in the trajectory an average loss $\bar{L}$ is calculated for the *model* (line 18 in Algorithm 3). From this loss, a confidence measure β is calculated, where low values of β denote high confidence. The β coefficient is an indicator of the performance of the base model and is used in the process of simulating the ensemble, i.e., combining the simulations of the constituents into an overall ensemble prediction.

4.2.2. Boosting of PBMs

The procedure for boosting of process-based models is presented in Algorithm 4. In analogy with bagging, the BOOSTING() procedure, takes the same four inputs: a library of domain knowledge *lib*, data consisting of training data D_T and validation data D_V, incomplete model *incompleteModel*, and an integer k denoting how many base models are to be generated. In contrast to bagging, here we start with the complete training data set (instead of a sample). In addition we introduce the concept of weights for each data/time-point, which are recalculated after every boosting iteration, (line 10 in Algorithm 4) according to the error made by the model from the previous iteration at the respective time point. In the learning phase, we use the WRMSE objective function, presented in Eq. (1).

The REWEIGHT() function takes 3 inputs: the highest ranked model (denoted with *model*) from the previous iteration, a data set D, and the respective set of weights ω. While this function resembles the CONFIDENCE() function, there are important differences: here we consider a set of time-point wise weights and loss (rather than a single overall loss), and we calculate this on the training data (in contrast to

Algorithm 4 Boosting process-based models.

```
1:  procedure BOOSTING($lib, \{D_T, D_V\}, incompleteModel, k$)
2:  returns $Ensemble$
3:      $Ensemble \leftarrow \emptyset$                      ▷ set of base models
4:      $\omega_t \leftarrow 1$                  ▷ $\omega_t$ is the weight of time
                                          point $t$, where $t = 0..N$
                                          and $N$ is the number of
                                          measurements in $D_T$
5:      for $i = 1$ to $k$ do
6:          $modelList_i \leftarrow$ PROBMOT($lib, \{D_T, \omega\}, incompleteModel$)
7:          $bestModel_i \leftarrow$ RANK($modelList_i, D_V$)
8:          $\beta_i \leftarrow$CONFIDENCE($bestModel_i, D_V$)
9:          $Ensemble \leftarrow Ensemble \bigcup (bestModel_i, \beta_i)$
10:         $\omega \leftarrow$ REWEIGHT($model_i, D_T, \omega$)
11:     end for
12: end procedure
13:
14: function REWEIGHT($model, D, \omega$)   returns $\omega$
15:     let $\hat{y}$                       ▷ simulated system variable $y$
16:     let $y$                             ▷ measured system variable $y$
17:     $\hat{y} \leftarrow$SIMULATE($model, D$)
18:     $maxDisc \leftarrow \sup(|y_t - \hat{y}_t|)^2$     ▷ calculate max discrep-
                                          ancy between measure-
                                          ments $y$ and simulation
                                          $\hat{y}$, where $t = 0..N$ and $N$ is
                                          number of time-points in
                                          D
19:     $L_t \leftarrow \frac{|y_t - \hat{y}_t|^2}{maxDisc}$          ▷ calculate square loss at each
                                          time point $t$, where $t = 0..N$
20:     $\bar{L} \leftarrow \sum_{t=0}^{N} L_t * \frac{\omega_t}{\sum_{t=0}^{N} \omega_t}$   ▷ calculate weighted average loss,
                                          according to the weights
21:
22:     $\omega_t \leftarrow \omega_t * \left[\frac{\bar{L}}{1-\bar{L}}\right]^{1-L_t}, t = 0..N$     ▷ update weights
23:     $\omega \leftarrow$ NORMALIZE($\omega, N$)        ▷ normalize weights to $N$
24: end function
```

validation data). First, the *model* is simulated on the data set D. Next, based on the error at each time point in the trajectory and the set of weights ω, the weighted average loss $\bar{L}$ is calculation. Finally, the set of weights is updated: the smaller the loss the more the weight is reduced – focusing on harder parts of the data set in the future iterations of the algorithm.

The output of the BOOSTING() procedure is a set of pairs (process-based models and their respective confidences) denoted with *Ensemble*. The highest ranked process-based model from each boosting iteration is considered as an ensemble constituent, for which a confidence is calculated. As in bagging, the ranking can be based on the performance of the process-based model on a separate validation data set D_V, or on the training sample (if $D_V == D_T$).

5. Experimental setup

In this section, we present the setup of the experiments used to evaluate the predictive performance of the ensembles of process-based models. The aim of the evaluation presented is three-tiered. First, we want to identify the ensemble method that leads to best long-term predictions of process-based models. In particular, we perform a comparative analysis of the predictive accuracies of the models obtained with bagging and boosting to the predictive accuracy of a single model. These experiments will confirm our hypothesis that ensembles of process-based models improve over the predictive performance of single models. Second, for each of the methods considered, we identify the optimal design decisions in terms of choosing the ensemble constituents, combining their predictions and choosing the ensemble size. In the following sections, we first introduce the data sets to be used in the experiments, then briefly describe the two other ProBMoT inputs, i.e., the library of modeling knowledge and incomplete models, and finally define the performance metrics used to asses the process-based models and ensembles thereof.

5.1. Data sets

In the experiments, we use fifteen data sets from the domain of aquatic ecosystems, in particular, modeling food-web dynamics in the three lakes of Bled in Slovenia, Kasumigaura in Japan, and Zurich in Switzerland. The original data sets comprise monthly measurements in the seven year periods from 1986 to 1992 for the Lake Kasumigaura (Atanasova et al., 2006a) and in the period from 1996 to 2002 for Lake Bled (Atanasova et al., 2006c) and Lake Zurich (Dietzel, Mieleitner, Kardaetz, & Reichert, 2013). To obtain daily values, the measurements were interpolated and daily samples were taken from the interpolation.

For each aquatic ecosystem, we split the original multi-year data sets into seven single-year data sets. Five of these were used (one at a time) for training the base-level models, one was used for validating the models in the process of selecting the ensemble constituents, and one was used to measure the predictive performance of the learned process-based models and ensembles thereof. Thus, we perform fifteen learning experiments; in each, we take a single-year training data set, learn a model using the train and the validation data sets, and test its predictive performance on the test data set. In the tables reporting the experimental results, we label the experiments with the labels B1–B5, K1–K5 and Z1–Z5 corresponding to the training data set using in the experiment, where, e.g., K3 denotes the Lake Kasumigaura data set for the third year (i.e., 1988).

5.2. ProBMoT inputs and parameter settings

In our experiments, we use the library of domain-specific knowledge for modeling population dynamics in aquatic ecosystems, based on the previous work by Atanasova et al. (2006b). To reduce the computational complexity of the experiments performed in this paper, we used a simplified version of the library, where we omitted some of the alternatives for modeling individual processes. The simplified version of the library and the incomplete models, lead to 320 candidate model structures (as opposed to 18144 with the whole library) for Lake Kasumigarua and 128 candidates (as opposed to 27216 with the whole library) for the other two lakes.

We use the same structure of the population dynamics model for all three aquatic ecosystems. It includes a single ordinary differential equation for a system variable representing the phytoplankton biomass (measured as *chlorophyll-a* in Lake Kasumigaura). The exogenous variables include the concentration of zooplankton *Daphnia hyalina* (where available, i.e., for Bled and Zurich only), the dissolved inorganic nutrients of nitrogen, phosphorus, and silica (ammonia in Kasumigaura), as well as two input variables representing the environmental influence of water temperature and global solar radiation (light).

ProBMoT uses the Differential Evolution (Storn & Price, 1997) method for parameter estimation with population size 50, strategy *rand/1/bin*, differential weight (F) and crossover probability (Cr) of 0.6. The limit on the number of evaluations of the objective function is one thousand per parameter. For simulating the ODEs, we used the CVODE simulator with absolute and relative tolerances set to 10^{-3}.

5.3. Evaluation metrics

To evaluate the predictive performance of a given model m, we use the relative root mean squared error (*ReRMSE*) (Breiman, 1984), defined as:

$$ReRMSE(m) = \sqrt{\frac{\sum_{t=0}^{N}(y_t - \hat{y}_t)^2}{\sum_{t=0}^{N}(\bar{y} - \hat{y}_t)^2}}, \tag{2}$$

where N denotes the number of measurements in the test data set, y_t and $\hat{y}_t$ correspond to the *measured* and *predicted*[2] value of the system variable y at time point t, and $\bar{y}$ denotes the mean value of y in the test data set. Note that the usual root mean squared error is considered here relative to the standard deviation of the system variable in the test data, thus allowing us to compare the errors of models for system variables measured on different scales.

We observe and compare the predictive performance (*ReRMSE*) of the models learned using different algorithms on the 15 data sets. To assess the statistical significance of the differences in performances among models obtained with different algorithms, we follow the recommendation by Demšar (2006) and use the corrected (Iman & Davenport, 1980) Friedman test (Friedman, 1940), followed by the post-hoc Nemenyi test (Nemenyi, 1963). The Nemenyi test is used to explain where the significant differences come from (from which pairs of algorithms) by computing the critical distance between the algorithm ranks at a given significance level; in our case, we set the significance level threshold at 95% (i.e., $p = 0.05$). The results of the Friedman–Nemenyi tests are depicted in average rank diagrams, such as the ones in Fig. 3.

Finally, to explore the correlation between ensemble performance and the diversity of the ensemble constituents, we measure the diversity of the constituents of the ensemble e as the average pairwise distance between the simulations of the constituents:

$$Diversity(e) = \frac{1}{\binom{|e|}{2}} \sum_{\{m_1,m_2\}\subset e} \sqrt{\frac{\sum_{t=0}^{N}(y_{1,t} - y_{2,t})^2}{N}}, \tag{3}$$

where $|e|$ denotes the number of models in the ensemble, N the number of measurements in the data set, m_1 and m_2 two models from e, and $y_{1,t}$ and $y_{2,t}$ the simulated states of these two models at time point t. To assess the performance improvement of the ensemble e over a single model m, we calculate:

$$Improvement(e, m) = -\frac{ReRMSE(e) - ReRMSE(m)}{ReRMSE(m)}. \tag{4}$$

We draw a scatter plot that depicts the correlation between ensemble diversity and performance improvement and calculate the Pearson correlation coefficient between them.

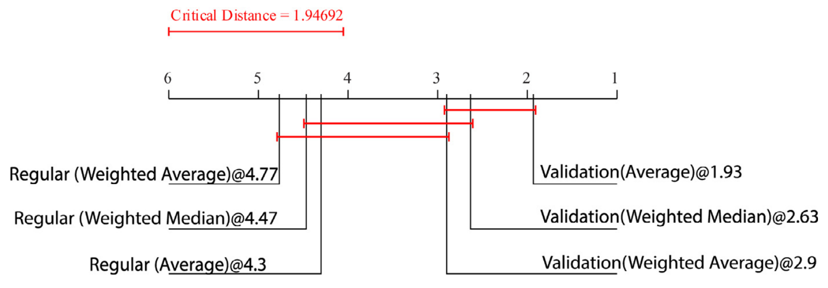
6. Results

In this section, we present and discuss the results of the empirical evaluation. In particular, we first identify the most suitable design decisions for individual ensemble methods. Then, we test the validity of our central hypothesis that ensembles of process-based models outperform the single process-based models. Finally, we investigate whether the performance improvement is related to the diversity of the predictions obtained with the ensemble constituents. Finally, we discuss the results in the context of the related machine learning research on ensembles.

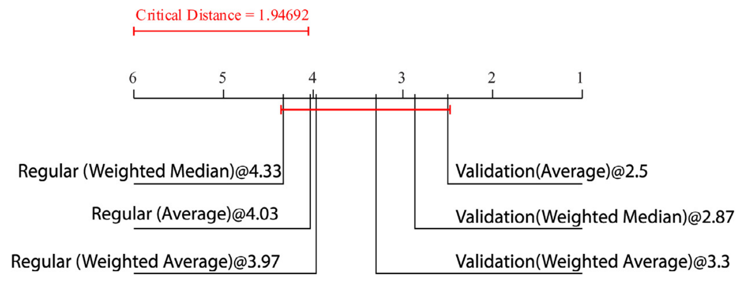
6.1. Learning ensembles of process-based models

The first design decision in an algorithm for learning ensembles of PBMs is related to the way of choosing the ensemble constituents. In each iteration of learning an ensemble, we select a single ensemble constituent, i.e., the highest ranked model from the ProBMoT output. Note that the latter represents a list of models, ranked with respect

[2] Predictions are obtained by simulating the model m on a test set.



(a) Bagging



(b) Boosting

Fig. 3. Comparison of the average ranks of different methods for selecting (and combining) ensemble constituents in terms of the predictive model performance averaged over the fifteen data sets in the case of bagging (top) and boosting (bottom). The labels describe the methods and give their average ranks.

to their performance on the training set. We refer to this base-line method for choosing ensemble constituents as *regular*. Alternatively, to avoid overfiting, we can re-rank the ProBMoT output, i.e., list of models, according to their performance on a separate validation data set. We refer to this selecting method as *validation*.

Fig. 3 summarizes the comparison between the performance of the regular and the validation method for choosing the base models to be included in the ensemble built in 50 iterations. The upper diagram depicts the results of the Friedman-Nemenyi test in the case of bagging, while the lower diagram depicts the results for the case of boosting. In both cases, choosing the base models based on their ranking on a separate validation set leads to superior ensemble performance. In the case of bagging, the superiority is statistically significant (note that the critical distance is smaller than the difference between the best validation rank and the best regular rank), while in the case of boosting it is not statistically significant.

We conjectured earlier that choosing ensemble constituents based on their performance on the training data set leads to ensembles that overfit the training data. Fig. 4 confirms the validity of this conjecture: in both cases (bagging and boosting), ensembles built using the regular selection method outperform the ones built with the validation selection method. This demonstrates a clear case of overfitting — while being superior on the training data, the regular selection method leads to ensembles with poor predictive performance.

Next, we focus on the design decision concerning the most appropriate method for combining the simulations of the base models in the ensemble. We compare the performance of three methods commonly used in learning ensembles of regression models: average, weighted average, and weighted median (Breiman, 1984; Drucker, 1997). Fig. 5 depicts the comparison of the average ranks of the three methods for combining the base-model simulations in the case of bagging and boosting with 50 iterations in each case. In both cases, the simplest method, i.e. *average*, outperforms the other two. In the case of bagging, the observed difference between the average and the weighted average methods is statistically significant, while the other differences are not significant. Given this, and following the parsimony principle, we can conclude that the most appropriate method for combining the simulations of the ensemble constituents is the simple average.

For all experiments so far, we learned ensembles of fixed size, always consisting of 50 base models. In the last series of experiments, we aim at making a decision on the optimal number of iterations for learning ensembles of process-based models. To this end, we compare the predictive performance of ensembles consisting of 5, 10, 25 and 50 base models. Fig. 6 summarizes the results of the comparison for both bagging and boosting. The Friedman-Nemenyi diagram shows that the ensemble built in 25 iterations leads to the best performance in both cases. Note however, that the observed differences in performance are not statistically significant.

In summary, based on the presented results we make the following design decisions related to learning ensembles of process-based models: We choose the ensemble constituents based on their performance on a separate validation data set, we combine the base-model predictions (simulations) using simple, unweighted, average and perform 25 iterations of adding base-models to the ensemble. In all the further experiments, we used these algorithm settings for learning ensembles.

6.2. Ensemble performance and diversity

Having made the design decisions for learning ensembles of process-based models, we necessary now focus on testing our central hypothesis that ensembles improve the predictive performance of process-based models. To this end, we compare the predictive performance of the ensembles with the one of a single process-based model learned on the whole training data set and chosen based on its performance on the separate validation data set. Fig. 7 depicts the

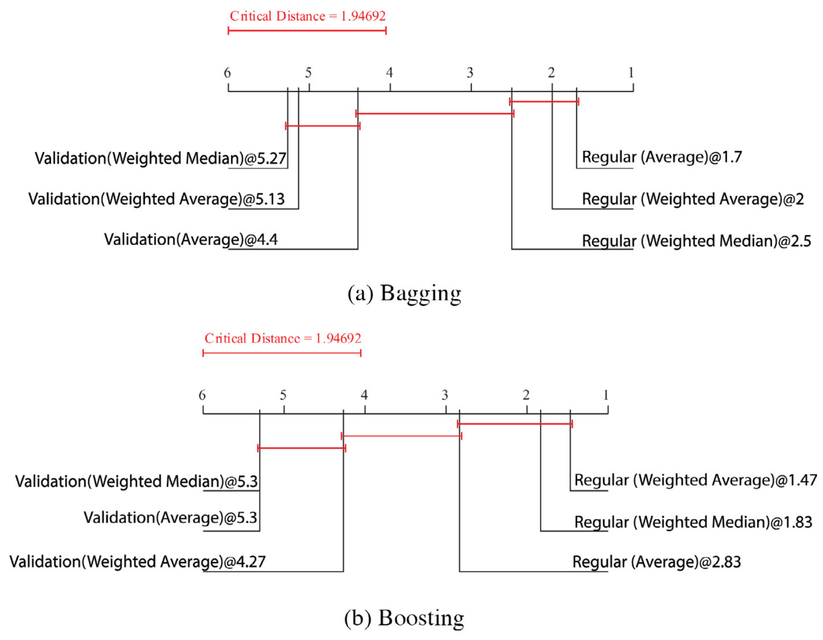


Fig. 4. Comparison of the average ranks of different methods for selecting (and combining) ensemble constituents in terms of the descriptive model performance averaged over the fifteen data sets in the case of bagging (top) and boosting (bottom). The labels describe the methods and give their average ranks.

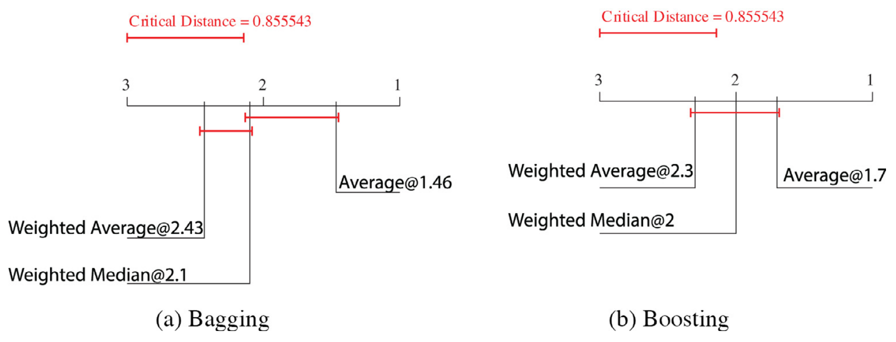


Fig. 5. Comparison of the average ranks of the three methods for combining the simulations of base models (average, weighted average, and weighted median) in terms of predictive performance averaged over the fifteen data sets in the case of bagging (left-hand side) and boosting (right-hand side).

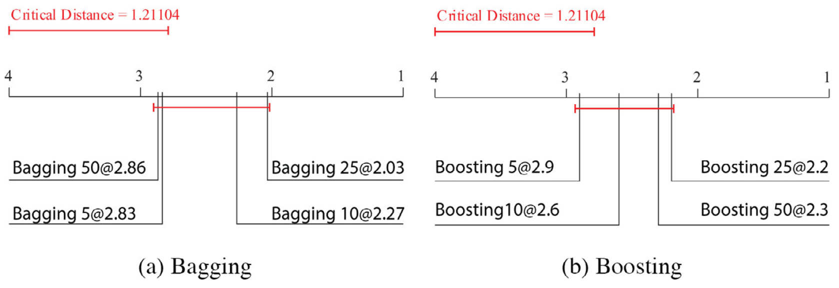


Fig. 6. Comparison of the average ranks of ensembles that include 5, 10, 25, and 50 base models in terms of predictive performance averaged over the fifteen experimental data sets.

comparison of the average ranks of a single model as well as the bagging and boosting ensembles, averaged over fifteen data sets.

The results of the Friedman-Nemenyi test show that both ensemble methods outperform the single process-based models. More importantly, the bagged ensembles *significantly* outperform the single models. These results support the central hypothesis of our paper that the ensembles improve the predictive performance of process-based models.

In our last set of experiments, we explore the relation of the observed significant improvement to the diversity of the simulations of the ensemble constituents. We first measure the relative improvement of the performance obtained by using an ensemble instead of a

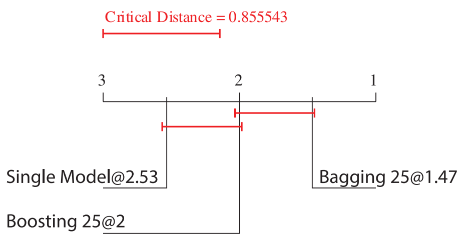


Fig. 7. Comparison of the average ranks of the single model and the ensembles (learned by using bagging and boosting) in terms of predictive performance averaged over the fifteen data sets.

Table 4
Diversity of the base models and the percentage of the relative improvement of the ensemble error over the error of the single model for the fifteen data sets. The Pearson correlation coefficient *r* between the improvement and the diversity is also given.

Case	Bagging		Boosting	
	Diversity	Improvement	Diversity	Improvement
B1	0.354	12.27%	0.342	12.97%
B2	0.561	4.48%	0.729	16.69%
B3	0.230	9.24%	0.477	11.77%
B4	0.617	17.45%	0.919	14.12%
B5	0.270	6.81%	0.408	24.29%
K1	1.010	1.04%	0.827	−24.32%
K2	1.030	35.06%	1.319	25.38%
K3	0.543	9.45%	0.656	3.59%
K4	0.598	1.02%	0.759	9.64%
K5	0.605	−1.53%	0.737	−7.97%
Z1	0.089	0.69%	0.411	−13.70%
Z2	0.234	7.23%	0.585	6.25%
Z3	0.223	5.72%	0.317	−1.01%
Z4	0.125	−2.33%	0.157	−4.21%
Z5	0.285	32.69%	0.702	15.16%
r	0.274		0.261	

single model. Then, we measure the diversity of base models in the ensemble. Finally, we analyze the correlation between the two.

Table 4 and Fig. 8 summarize the results of these experiments. The results presented in Table 4 confirm our previous finding: bagging outperforms single models in all but two data sets, K5 and Z4. Note that the loss of performance for these two data sets is minor (below 3%). On the other hand, the gain in performance can be substantial and reach up to 17% for Lake Bled, 35% for Lake Kasumigaura and 33% for Lake Zurich.

In the case of boosting, the improvement over the single model is less substantial, and more importantly, less consistent. The boosting method outperforms the single model for the majority of the training data sets (up to 25% for the case of Lake Bled; up to 26% for Lake Kasumigaura; and 16% for Lake Zurich). However for the remaining five data sets (K1, K5, Z1, Z3 and Z4) it under-performs. Note that two of these (K5 and Z4) are the same as the ones where bagging under-performs. For the remaining three cases, bagging makes a modest (up to 6%) improvement over the single model. This confirms that bagging is a better method for learning ensembles of process-based models.

Finally, we observe a varying degree of diversity between ensemble constituents for different data sets—diversity varies from 0.125 to 1.030. The scatter plots in Fig. 8 show weak positive correlation between ensemble diversity and relative improvement of performance. While measured Pearson correlation coefficient of 0.274 for bagging, and 0.261 for boosting, is neither high nor significant, the positive correlation is in line with the implicit assumption that ensembles improve predictive performance by exploiting the diversity of their constituents (Kuncheva & Whitaker, 2003).

6.3. *Discussion*

The results presented in this paper confirm our main hypothesis that ensembles of process-based models yield a significant gain in predictive performance when compared to a single model. Based on the performed empirical evaluation, we identified the main design decisions that need to be made when learning such ensembles by using bagging and boosting as underlying methods. In this context, it is very important that one uses a separate validation data set in addition to the training one when learning the base models included in the ensemble. The optimal ensembles of PBMs consist of relatively low numbers of constituent models, ranging between 10 and 25 for bagging and 25–50 for boosting (for both methods, the best performing ensembles comprised 25 constituents). For combining the simulation of the constituent process-based models, the results showed that the simplest combining scheme, i.e averaging, provides the most satisfying results both in terms of predictive accuracy and computational complexity.

The process-based models, when simulated in a predictive setting, can often produce divergent simulations, i.e., simulations where the systems variables leave their plausible ranges. Therefore, when simulating ensembles of PBMs, we explicitly handle this kind of behavior of the base models. We use the provided domain knowledge on system variable ranges to discard the invalid behaviors from the

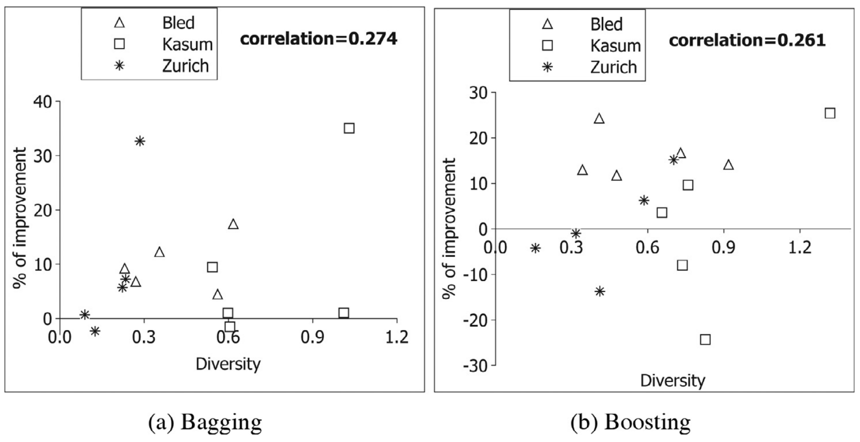


Fig. 8. Scatter plots depicting the correlation between the diversity of the base-model predictions and the relative error improvement between a single model and an ensemble for fifteen data sets.

resulting ensemble prediction. This can be viewed as a dynamic form of ensemble pruning.

Finally, our major conclusion is that both methods for learning ensembles of process-based models, following the design outlined above, outperform single models. More importantly, bagged process-based models provide a significant performance gain over a single model. Note that the improvement of performance over the single model is positively related to the diversity of the ensemble constituents — the higher the diversity, the greater the improvement.

However, in our case, the correlation between the diversity and the performance gain is very weak. Breiman (1996a) states that bagging can improve predictive performance when the ensemble is composed of models whose predictions vary sufficiently. The base models obtained in our approach have only modest diversity, which may limit the full predictive potential of the ensembles. One reason for this may be the lower number of model structures considered by ProBMoT, which used a simplified library of domain knowledge. This can be overcome by using the original library, which leads to tens of thousands of model structures.

In addition, the study of Joshi, Agarwal, and Kumar (2002) points out that the performance of boosting is correlated with the performance of the base learner, in our case, ProBMoT. This means that, when a single model obtained with ProBMoT, exhibits very good predictive performance, the ensembles exhibit similar or worse performance. This may be one reason for the behavior seen in Table 4 for the five highlighted experimental data sets (K1, K5, Z1, Z3 and Z4). Further investigations are required to determine whether increased diversity could lead to better performance.

7. Conclusion

In this paper, we address the task of learning ensembles of process-based models by designing, implementing and evaluating appropriate methodology. The developed methodology is general enough to allow for adapting different ensemble methods to the particular context of learning process-based models. This methodology is the main contribution of our paper, since it extends the scope of current process-based modeling approaches to the task of learning ensembles of process-based models. While the methodology has been only used in the limited context of adapting bagging and boosting, we can easily extend it towards other methods for learning homogeneous ensembles.

The second contribution of our paper is the extension of the scope of ensemble methods to process-based modeling. While previous proof-of-concept experiments have been performed for specific types of ensembles (Simidjievski et al., 2015), this is the first paper to provide a detailed layout of a methodology for building various types of ensembles of process-based models. This contribution is also important in the wider context of ensembles for time-series forecasting (Kourentzes et al., 2014; Ma et al., 2015; Tay et al., 2013). While forecasting ensembles have a narrow focus on short-term prediction tasks, where the value of the time series at the next time point is predicted, ensembles of process-based models provide accurate long-term predictions over many future time points. In contrast to Bridewell et al. (2005), who build ensembles that explain observed (albeit long-term) system behavior, the methods presented here provide accurate predictions of the unobserved future system behavior.

Note also that the results of the performed experimental evaluation confirm our central conjecture that ensembles provide much more accurate predictions of future concentrations of species in an aquatic ecosystems than a single process-based model. While single models struggle with achieving the performance of the baseline predictor (that predicts constant species concentrations at the level of their average), the ensembles of process-based models lead to accurate predictions of population dynamics over a long prediction periods, e.g., one season (year) in advance. When compared to previous results obtained in the domain of population dynamics (Atanasova et al., 2006a, 2006c), this is also a non-trivial improvement of predictive performance over the state-of-the-art models of population dynamics. These results are consistent over experiments with data from three real-world aquatic ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan, and Lake Zurich in Switzerland. This is the third important contribution of our paper, which mainly contributes to the domain of ecological modeling.

Several directions for further work can be followed. First, note that the validity of the results presented in this paper is limited to the particular domain of modeling population dynamics in aquatic ecosystems. An immediate continuation of the work presented here is to investigate the generality of the results across various domains and modeling tasks: both the superior performance of ensembles of process-based models (as compared to individual models) and the optimal settings and design decisions for the algorithms for learning them need to be verified for other domains and datasets. Next, following ideas from Bridewell et al. (2005), we intend to explore methods for incorporating the structure of the ensemble constituents into a single process-based model with good predictive performance.

Here we limit our attention to a single type of ensembles, where the diversity in the ensemble is obtained by learning ensemble constituents from different samples of the training data. In future, we can extend this narrow scope by considering other methods for generating ensemble constituents that rely on sampling the model variables or sampling the model components/templates in the library of modeling knowledge. The first method directly relates to the standard ensemble method of random subspaces (Ho, 1998). The second method would take different samples of entity and process templates from the library when learning individual models with the extra benefit of reducing the computational complexity of the individual learning tasks due to the reduced complexity of the search space.

Finally, we intend to extend our methodology towards learning interactive ensembles of models of dynamic systems, referred to as super-models (Mirchev, Duane, Tang, & Kocarev, 2012; van den Berge, Selten, Wiegerinck, & Duane, 2011). In contrast to ensembles, where the base models are learned and simulated independently and combined afterwards, within super-models, the base models can share and interchange information both during the learning and the simulation phase. In this context, one can learn the constituent models, their interconnections or both.

Acknowledgments

We thank Pančе Panov and Dragi Kocev for the helpful discussions and comments on the work presented in this paper. We also thank Nataša Atanasovska for the knowledge and early work on the systems of Lake Bled and Lake Kasumigaura, along with Anne Dietzel for data from and knowledge about Lake Zurich. We would also like to acknowledge the financial support of the Slovenian Research Agency, via the grant P2-0103 and a young researcher grant to the first author, as well as the European Commission, via the grants ICT-2013-612944 MAESTRA and ICT-2013-604102 HBP.

References

Atanasova, N., Recknagel, F., Todorovski, L., Džeroski, S., & Kompare, B. (2006a). Computational assemblage of ordinary differential equations for chlorophyll-a using a lake process equation library and measured data of Lake Kasumigaura. In F. Recknagel (Ed.), *Ecological Informatics* (pp. 409–427). Springer.
Atanasova, N., Todorovski, L., Džeroski, S., & Kompare, B. (2006b). Constructing a library of domain knowledge for automated modelling of aquatic ecosystems. *Ecological Modelling, 194*(13), 14–36.
Atanasova, N., Todorovski, L., Džeroski, S., Remec, R., Recknagel, F., & Kompare, B. (2006c). Automated modelling of a food web in Lake Bled using measured data and a library of domain knowledge. *Ecological Modelling, 194*(1-3), 37–48.
Breiman, L. (1984). *Classification and regression trees*. Chapman & Hall.

Breiman, L. (1996a). Bagging predictors. *Machine Learning, 24*(2), 123–140.
Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32. doi:10.1023/A:1010933404324.
Bridewell, W., Asadi, N. B., Langley, P., & Todorovski, L. (2005). Reducing overfitting in process model induction. In *Proceedings of the 22nd international conference on machine learning (ICML '05)* (pp. 81–88). ACM Press.
Bridewell, W., Langley, P. W., Todorovski, L., & Džeroski, S. (2008). Inductive process modeling. *Machine Learning, 71*, 1–32.
Čerepnalkoski, D., Taškova, K., Todorovski, L., Atanasova, N., & Džeroski, S. (2012). The influence of parameter fitting methods on model structure selection in automated modeling of aquatic ecosystems. *Ecological Modelling, 245*, 136–165.
Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 7*, 1–30.
Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Proceedings of the 1st international workshop on multiple classifier systems* (pp. 1–15). Springer.
Dietzel, A., Mieleitner, J., Kardaetz, S., & Reichert, P. (2013). Effects of changes in the driving forces on water quality and plankton dynamics in three Swiss lakes long-term simulations with BELAMO. *Freshwater Biology, 58*(1), 10–35. doi:10.1111/fwb.12031.
Drucker, H. (1997). Improving regressors using boosting techniques. In *Proceedings of the 14th international conference on machine learning (ICML'97)* (pp. 107–115). Morgan Kaufmann Publishers Inc.
Džeroski, S., & Todorovski, L. (2003). Learning population dynamics models from data and domain knowledge. *Ecological Modelling, 170*, 129–140.
Freund, Y. (1999). An adaptive version of the boost by majority algorithm. In *Proceedings of the 12th annual conference on computational learning theory (COLT'99)* (pp. 102–113). ACM Press.
Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139.
Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics, 11*(1), 86–92. doi:10.2307/2235971.
Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(8), 832–844.
Iman, R. L., & Davenport, J. M. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics - Theory and Methods, 9*(6), 571–595. doi:10.1080/03610928008827904.
Jiménez, A., Barba, I., del Valle, C., & Weber, B. (2013). Optbpplanner: automatic generation of optimized business process enactment plans. In H. Linger, J. Fisher, A. Barnden, C. Barry, M. Lang, & C. Schneider (Eds.), *Building sustainable information systems* (pp. 429–442). US: Springer. doi:10.1007/978-1-4614-7540-8_33.
Jørgensen, S. E., & Bendoricchio, G. (2001). *Fundamentals of ecological modelling*: vol. 21. Elsevier.
Joshi, M. V., Agarwal, R. C., & Kumar, V. (2002). Predicting rare classes: can boosting make any weak learner strong? In *Proceedings of the 8th ACM sigkdd international conference on knowledge discovery and data mining (KDD'02)* (pp. 297–306). ACM Press. doi:10.1145/775047.775092.
King, M. A., Abrahams, A. S., & Ragsdale, C. T. (2014). Ensemble methods for advanced skier days prediction. *Expert Systems with Applications, 41*, 1176–1188.
Kourentzes, N., Barrow, D. K., & Crone, S. F. (2014). Neural network ensemble operators for time series forecasting. *Expert Systems with Applications, 41*(9), 4235–4244. http://dx.doi.org/10.1016/j.eswa.2013.12.011.
Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning, 51*(2), 181–207.
Luenberger, D. (1979). *Introduction to dynamic systems: theory, models, and applications*. Wiley.
Ma, Z., Dai, Q., & Liu, N. (2015). Several novel evaluation measures for rank-based ensemble pruning with applications to time series prediction. *Expert Systems with Applications, 42*(1), 280–292. http://dx.doi.org/10.1016/j.eswa.2014.07.049.
Maclin, R., & Opitz, D. (1999). Popular ensemble methods: an empirical study. *Journal of Artificial Intelligence Research, 11*, 169–198.
Mirchev, M., Duane, G. S., Tang, W. K., & Kocarev, L. (2012). Improved modeling by coupling imperfect models. *Communications in Nonlinear Science and Numerical Simulation, 17*(7), 2741–2751.
Nemenyi, P. B. (1963). *Distribution-free Multiple Comparisons* Ph.D. thesis. Princeton University.
Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33*(1-2), 1–39. doi:10.1007/s10462-009-9124-7.
Simidjievski, N., Todorovski, L., & Džeroski, S. (2015). Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems. *Ecological Modelling, 306*(0), 305–317. http://dx.doi.org/10.1016/j.ecolmodel.2014.08.019.
Smith, D., Dutta, R., Hellicar, A., Bishop-Hurley, G., Rawnsley, R., Henry, D., et al. (2015). Bag of class posteriors, a new multivariate time series classifier applied to animal behaviour identification. *Expert Systems with Applications, 42*(7), 3774–3784. http://dx.doi.org/10.1016/j.eswa.2014.11.033.
Storn, R., & Price, K. (1997). Differential evolution a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*(4), 341–359. doi:10.1023/A:1008202821328.
Taškova, K., Šilc, J., Atanasova, N., & Džeroski, S. (2012). Parameter estimation in a nonlinear dynamic model of an aquatic ecosystem with meta-heuristic optimization. *Ecological Modelling, 226*, 36–61.
Tay, W.-L., Chui, C.-K., Ong, S.-H., & Ng, A. C.-M. (2013). Ensemble-based regression analysis of multimodal medical data for osteopenia diagnosis. *Expert Systems with Applications, 40*(2), 811–819.
Todorovski, L., Bridewell, W., Shiran, O., & Langley, P. W. (2005). Inducing hierarchical process models in dynamic domains. In *Proceedings of the 20th national conference on artificial intelligence (NAI'05)* (pp. 892–897). AAAI Press.
Todorovski, L., & Džeroski, S. (2007). Integrating domain knowledge in equation discovery. In S. Dzeroski, & L. Todorovski (Eds.), *Computational discovery of scientific knowledge*. In *Lecture Notes in Computer Science: vol. 4660* (pp. 69–97). Springer.
van den Berge, L. A., Selten, F. M., Wiegerinck, W., & Duane, G. S. (2011). A multi-model ensemble method that combines imperfect models through learning. *Earth System Dynamics, 2*(1), 161–177.
White, S. A. (2004). *Process modeling notations and workflow patterns. Technical report*. IBM Corporation.
Wolpert, D. H. (1992). Stacked generalization. *Neural Networks, 5*, 241–259.

Chapter 6

Learning Ensembles via Sampling the Library of Domain Knowledge

In this chapter, we present the results of the empirical evaluation concerned with learning ensembles by sampling the library of domain knowledge, i.e., learning ensembles with the random library subsamples (RLS) method.

The experiment evaluation presented in the previous chapter showed that ensembles of process-based models obtained by using different samples of the measured data, i.e., bagging and boosting, yield significantly improved predictive performance. However, this improvement comes at the cost of a substantial increase in the computational complexity of learning the predictive ensemble model. To tackle this problem, the ensemble method proposed in this chapter, titled *"Modeling dynamic systems with efficient ensembles of process-based models"*, aims at efficiently learning ensembles of process-based models while maintaining their accurate long-term predictive performance.

More specifically, here we propose a process-based ensemble method which extends the existing equation discovery approaches for learning ensembles of processes-based models towards sub-sampling domain knowledge instead of the measurements. Similarly to the previous studies, we apply the proposed method and evaluate its performance on a set of problems of automated modeling of population dynamics in three real-life aquatic ecosystems. We first identify the most appropriate design decisions within the algorithm for learning such ensembles. We then compare their predictive performance to that of other ensembles of process-based models learned on different subsamples of the measurements as well as to the performance of an individual process-based model.

Based on the results of the empirical evaluation presented, we identified the most appropriate design choices when constructing ensembles by sampling the library of domain knowledge. First, when selecting the ensemble constituents, it is necessary to use separate validation data, in addition to the training data used for learning the models. This yields more robust ensembles, which in turn substantially improve the predictive performance. Second, the ensembles should consist of a relatively small number of base models (i.e., 10) and should use standard unweighted averaging for combining the predictions of the individual base models. Note that these findings support the optimal design decisions for learning ensembles of process-based models via bagging and boosting, i.e., learning ensembles with a relatively small number of constituents, chosen by using a separate validation set, and combined by averaging, yield the best performance.

To properly address all aspects of the RLS method, we propose two different alternatives, i.e., sampling the library with and without duplicates. The results of the experimental evaluation showed that ensembles constructed by sampling the library with duplicates have (not significantly) better predictive performance as compared to their counterpart without duplicates. Additionally, these two alternative types of ensembles showed better predictive performance as compared to both types of baseline ensembles, i.e, the one comprised of the highest-ranking models and the one comprised of random models. Given that, here we aim at *efficient* learning of ensembles and following the parsimony principle we conclude that ensembles constructed by RLS, which allow duplicates are computationally more efficient, while maintaining good predictive performance.

More importantly, the results presented in this study reconfirm our hypothesis that ensembles of process-based models, learned by sampling the library of domain knowledge, significantly outperform single models in terms of predictive performance. In comparison to ensembles of process-based models constructed with bagging and boosting, the RLS ensembles performed slightly worse and slightly better, respectively, but the differences in performance are not significant. On the other hand, the computational performance analysis showed that the library sampling learning method is 25 times faster compared to its bagging and boosting learning counterparts (with equal number of bagging/boosting iterations). This is a non-trivial improvement in computational efficiency as compared to the state-of-the art methods for learning ensembles of process-based models.

All things considered, the experiments show that the RLS ensembles yield to significantly more accurate predictions of population dynamics as compared to a single process-based models, while being substantially more efficient than the other methods for learning ensembles of process-based models.

RESEARCH ARTICLE

Modeling Dynamic Systems with Efficient Ensembles of Process-Based Models

Nikola Simidjievski[1,2]*, **Ljupčo Todorovski**[3], **Sašo Džeroski**[1,2]

1 Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia, **2** Jožef Stefan International Postgraduate School, Ljubljana, Slovenia, **3** Faculty of Administration, University of Ljubljana, Ljubljana, Slovenia

* nikola.simidjievski@ijs.si

OPEN ACCESS

Citation: Simidjievski N, Todorovski L, Džeroski S (2016) Modeling Dynamic Systems with Efficient Ensembles of Process-Based Models. PLoS ONE 11 (4): e0153507. doi:10.1371/journal.pone.0153507

Editor: Bin Liu, Harbin Institute of Technology Shenzhen Graduate School, CHINA

Received: November 3, 2015

Accepted: March 30, 2016

Published: April 14, 2016

Copyright: © 2016 Simidjievski et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Data Availability Statement: Third party data are available from the original authors of the respective studies: Lake Bled-Data are from the [Atanasova N, Todorovski L, Džeroski S, Remec R, Recknagel F, Kompare B. Automated modelling of a food web in Lake Bled using measured data and a library of domain knowledge. Ecological Modelling. 2006;194 (1-3):37–48. DOI:10.1016/j.ecolmodel.2005.10.029] study whose authors may be contacted at: Nataša Atanasova -http://www.fgg.uni-lj.si/izh/atanasova; Špela Remec-Rekar -http://www.arso.gov.si/en Lake Kasumigaura- Data are from the [Atanasova N, Recknagel F, Todorovski L, Džeroski S, Kompare B. Computational Assemblage of Ordinary Differential

Abstract

Ensembles are a well established machine learning paradigm, leading to accurate and robust models, predominantly applied to predictive modeling tasks. Ensemble models comprise a finite set of diverse predictive models whose combined output is expected to yield an improved predictive performance as compared to an individual model. In this paper, we propose a new method for learning ensembles of process-based models of dynamic systems. The process-based modeling paradigm employs domain-specific knowledge to automatically learn models of dynamic systems from time-series observational data. Previous work has shown that ensembles based on sampling observational data (i.e., bagging and boosting), significantly improve predictive performance of process-based models. However, this improvement comes at the cost of a substantial increase of the computational time needed for learning. To address this problem, the paper proposes a method that aims at efficiently learning ensembles of process-based models, while maintaining their accurate long-term predictive performance. This is achieved by constructing ensembles with sampling domain-specific knowledge instead of sampling data. We apply the proposed method to and evaluate its performance on a set of problems of automated predictive modeling in three lake ecosystems using a library of process-based knowledge for modeling population dynamics. The experimental results identify the optimal design decisions regarding the learning algorithm. The results also show that the proposed ensembles yield significantly more accurate predictions of population dynamics as compared to individual process-based models. Finally, while their predictive performance is comparable to the one of ensembles obtained with the state-of-the-art methods of bagging and boosting, they are substantially more efficient.

Introduction

Models are vital instruments for investigating how constitutive elements interact in the complex dynamic systems observed in nature. Scientists incorporate expert knowledge and statistical methods to recreate the observed behavior and find patterns in the measured data, which in

Equations for Chlorophyll-a Using a Lake Process Equation Library and Measured Data of Lake Kasumigaura. Ecological Informatics. Springer; 2006. p. 409–427. DOI: 10.1007/3-540-28426-5_20] study whose authors may be contacted at: Nataša Atanasova-http://www.fgg.uni-lj.si/izh/atanasova; Friedrich Recknagel http://www.adelaide.edu.au/directory/friedrich.recknagel. Lake Zurich-Data are from the [Dietzel A, Mieleitner J, Kardaetz S, Reichert P. Effects of changes in the driving forces on water quality and plankton dynamics in three Swiss lakes–long-term simulations with BELAMO. Freshwater Biology. 2013;58(1):10–35. DOI:doi:10.1111/fwb.12031] study whose authors may be contacted at anne.dietzel@eawag.ch.

Funding: NS received funding from the Slovenian Research Agency (https://www.arrs.gov.si, Grant P2-0103) and the European Commission (http://ec.europa.eu, Grant ICT-2013-604102 HBP). LT received funding from the Slovenian Research Agency (https://www.arrs.gov.si, Grant P5-0093(B)). SD received funding from the Slovenian Research Agency (https://www.arrs.gov.si, Grant P2-0103) and the European Commission (http://ec.europa.eu, Grants ICT-2013-612944 MAESTRA and ICT-2013-604102 HBP). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

turn results in a model of a dynamic system. The model can be simulated- both to describe and to predict its states under various conditions.

The process-based modeling paradigm, employed in this paper, follows the above principles by automatically modeling the behavior of dynamic systems using time-series measurements and domain-specific modeling knowledge. The resulting process-based models offer both an understandable formalism of the system's structure and a mathematical formulation allowing for its simulation. The former provides a high-level representation of the modeled system in terms of entities and processes, i.e., the elements in the system and their interactions, respectively. The latter, transforms these entity/process components into a system of ordinary differential equations (ODEs). ODEs are a widely accepted modeling formalism. They allow for long-term simulation of dynamic systems, which requires a minimal input consisting the initial value of the system's state and data corresponding to the system's exogenous variables.

Often, establishing models that provide an accurate prediction of the behavior of a dynamic system, require an expert intervention, which is time consuming and expansive. In contrast, the process-based modeling paradigm allows for automated construction of such models [1–4]. While these automatically constructed models can be employed for long-term prediction, the respective studies mostly focus on constructing models that provide a detailed and precise description of the observed behavior. Unfortunately, the resulting models often fail to accurately predict the subsequent states of the system. In this paper, we focus on the task of improving the predictive performance of process-based models by constructing ensembles.

An ensemble model is a combination of predictive models, which is expected to lead to more accurate prediction than the one obtained with a individual model [5]. Ensembles are a standard approach in machine learning for improving the predictive performance of models. They are usually employed in the context of the learning tasks of classification and regression [6, 7] to address the problems of over-fitting, high dimensionality, or missing features in the training data. Recent studies by Simidjievski et al. [8, 9] have introduced ensembles in the context of process-based models. This has allowed more accurate long-term prediction of the behavior of dynamic systems. The ensembles yield a significant gain in predictive performance over the individual process-based models. However, this performance gain comes at the cost of increased computational complexity.

The main contribution of this paper is a novel efficient method for learning ensembles of process-based models, that still accurately predict long-term behavior of dynamic systems. The method learns ensembles by sampling the library of domain-knowledge in a manner similar to the standard ensemble learning method of random subspaces [10]. We conjecture that such ensembles will outperform individual process-based models in terms of their accuracy of long-term predicting the system's behavior, and will overcome the computational limitations of the existing methods for learning ensembles of process-based models based on data sampling. To test the validity of this conjecture, we perform an extensive empirical evaluation of the implemented method on the task of modeling and predicting population dynamics in aquatic ecosystems. This empirical evaluation will allow us to identify the most appropriate design decisions within the algorithm for learning such ensembles of process-based models. It will also allow us to compare the predictive performance of the new method to the performance of the existing methods of bagging and boosting of process-based models.

The remainder of the paper is organized as follows. In the next section, we provide an overview of the related work from the areas of process-based modeling and ensemble learning. Section *Process-based models* introduces the process-based modeling paradigm and its latest implementation ProBMoT (Process-Based Modeling Tool), by illustrating its use on a simple modeling task from the domain of population dynamics. Section *Ensembles of process-based models* presents the extension of ProBMoT for learning ensembles of process-based models, focusing on learning ensembles by sampling the library of domain knowledge. Next, we present the experimental setup for evaluating the developed methods on three tasks of predictive modeling of population dynamics in aquatic ecosystems, i.e., the ecosystems of Lake Bled in Slovenia, Lake Kasumigaura in Japan and Lake Zurich in Switzerland. Section *Results* presents the findings of the empirical evaluation and gives a complexity analysis for learning ensembles of PBMs. In the next section, we discusses the results in the context of related research. Finally, the last section concludes the paper and suggests directions for further work.

Related research

The work presented in this paper builds upon our previous research on ensembles of processes-based models [8, 9]. These studies tackle the challenge of learning ensembles by sampling the training data. However, while the empirical evaluations in the respective studies have shown that ensembles of process-based models lead to significant gains in predictive performance, the process of learning such ensemble models is very computationally demanding.

In contrast, here we introduce a novel approach which will learn ensembles by sampling the library of domain knowledge. The work presented in this study, is closely related to the work performed by Bridewell et al. [11], where the authors report on generating an ensemble of PBMs by fusing the structures of the individual constituents. The empirical evaluation of the respective study shows that such ensembles have improved performance in a descriptive setting, i.e., in explaining the observed behavior, whereas their ability for long-term prediction (i.e., outside the scope of the training data time intervals) are not reported.

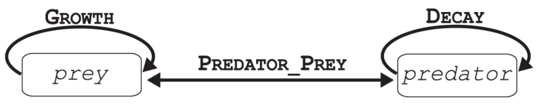
In a broader sense, this paper extends the scope of the state-of-the-art in equation discovery [12, 13], related to automatically obtaining descriptive models from domain knowledge. Our work comes closer to the inductive process modeling paradigm [14, 15] which mainly tackles the problem of automatically obtaining explanatory models of a dynamic system in a process-based representation. This paradigm has proven to be successful for a variety of modeling tasks of population dynamics in a number of real-world domains for explaining the observed behavior of the modeled system [1–4]. However, focusing on the provision of detailed and accurate descriptions of the observed systems, the models in the respective studies, have limited predictive abilities when applied to tasks of predicting subsequent system behavior.

Finally, this work is related to the long tradition of learning ensembles for tackling various predictive modeling tasks in different ecological domains. Kocev and Džeroski et al. [16] present an ensemble method for learning habitat models of communities of organisms under different environmental conditions, using predictive clustering trees [17]. However, the ensemble methods presented in this paper are most closely related to those that tackle the problem of time-series forecasting. Knudby et al. [18] present novel approaches for modeling fish-habitat relationships using support-vector machines and tree ensembles for regression, where they aim at predicting fish species richness, biomass, and diversity from a range of habitat variables. Their main contribution is the extensive empirical study which identifies the most suitable machine learning method for short-term prediction since it aims at forecasting the next-time-point of fish-community concentrations. In contrast, our process-based ensembles aim at long-term (typically one year) prediction of systems behavior that concern periods with potentially indefinite ranges of time points.

Materials and Methods

Process-based models

The models of dynamic systems aim to describe the activities of the system components and the change of the system states over time. Mathematical equations serve as a powerful tool for



A) Graphical representation of a Predator-Prey model

```
entity prey :  Population; {
    vars:  d {role endogenous; initial:20};}
entity predator :  Population; {
    vars:  d {role endogenous; initial:2};}
process GROWTH(prey):EXPONENTIALGROWTH
    { consts:  gR=2.5;}
process DECAY(predator):DECAY
    { consts:  dR=1.2;}
process PREDATOR_PREY(predator,prey):UNSATURATEDPP
    { consts:  eF = 0.1, iR = 0.3;}
```

B) Process-based modeling representation of a Predator-Prey model

Fig 1. A simple Predator-Prey model and its process-based modeling representation. (A) A graphical representation of the entities (white boxes) and processes (black arrows) in a simple Predator-Prey model and (B) its process-based modeling representation.

doi:10.1371/journal.pone.0153507.g001

achieving this aim, where the variables in the equations represent the state of the system components and the operators the interactions among components. Although this framework allows for adequate representation of dynamic systems, it forfeits the high-level information about the whys and hows of the modeled system's behavior.

In essence, process-based models provide a conceptualization of the structure of the observed system, accompanied by modeling details that allow for their transformation to equations and therefore simulation. They tackle the task of describing dynamic systems from two aspects: qualitative and quantitative. From a qualitative aspect, a process-based model is a set of entities and processes. The entities represent the components of the observed system, which are involved in activities represented by the processes. From a quantitative aspect, a process-based model is interpreted as a set of differential and/or algebraic equations which can be used to simulate the behavior of the observed dynamic system. Process-based models encode both this low-level quantitative mathematical formalism, and a high-level qualitative description of the system.

Fig 1 gives both a graphical representation (Fig 1A) and a qualitative process-based representation (Fig 1B) of an example Lotka—Volterra Predator–Prey model. Notice how the PBM formalism represents the different processes/relations and entities/components involved in the model. For example, the PREDATOR_PREY interaction is modeled as UNSATURATEDPP and involves two entities *predator* and *prey* which are both part of the *Population* involved in the modeled system. Moreover, the growth and the decay of the *predator* and *prey* are modeled as EXPONENTIALGROWTH and DECAY, respectively.

Table 1. Library of domain knowledge for modeling Predator-Prey dynamics.

```
template entity Population{
   vars: d{ aggregation:sum, unit:“kg/m3”; range:<0,500>} ;\\density
template process Growth(pop: Population) {
   consts: gR{ range:<0,5>}};
template process ExponentialGrowth:Growth{
   equations: td(pop.d) = gR * pop.d;}
template process LogisticGrowth:Growth{
   equations: td(pop.d) = gR * pop.d/(1 − pop.d/gR);}
template process Decay(pop: Population) {
   consts: dR{ range:<0,2>} ;
   equations: td(pop.d) = −dR * pop.d; }
template process Interaction(pop1:Population, pop2:Population){
   consts: iR{ range: <0,2>} , eF{ range: <0,1>} ;}
template process UnsaturatedPP:Interaction{
   equations: td(pop.d) = iR * eF * pop1.d * pop2.d,
              td(pop.d) = −iR * pop1.d * pop2.d; }
template process SaturatedPP:Interaction{
   consts: sR{ range: <0,10>} ;
   equations: td(pop.d) = iR * eF * pop1.d * pop2.d/(pop2.d+sR),
              td(pop.d) = −iR * pop1.d * pop2.d/(pop2.d+sR); }
```

doi:10.1371/journal.pone.0153507.t001

The entities and processes in Fig 1B represent the specific structure and dynamics of the particular system at hand. To elucidate such specific entities and processes, the process-based modeling approach uses entity and process templates, which serve as general placeholders for properties and definitions. The templates provide general modeling specifications for any instantiation to specific components or interactions, which in turn allows for a high-level qualitative conceptualization of a model to be translated into a low-level quantitative mathematical formalization which can be simulated.

Table 1, depicts the library of domain knowledge used to instantiate the Predator-Prey model from Fig 1B. The library organizes the templates in hierarchies. The *predator* and *prey* entities from the example are instances of the general template entity of *Population*. The entity templates incorporate group properties of the components of the modeled system. These properties include the variables (which change over time) and the constants related to the components of the modeled system, and their respective value constraints. For the variables, an aggregation function specifies how different process influences are combined for a particular entity. Take for example the *Population* template entity: it has one variable d which denotes density, and an aggregation function defined as summation, which means that, for the case of *prey*, the influences of both processes Growth and PREDATOR_PREY on the variable d will be summed.

The processes templates include specifications of the entity templates that interact, in terms of constants, algebraic and ordinary differential equations. The process templates are organized also into a hierarchy that defines the space of modeling alternatives. The particular process Growth, used in Fig 1B, is an instance of a more general process template Growth (Table 1), which is further instantiated to the process alternative— ExponentialGrowth (out of the two possible, i.e. ExponentialGrowth and LogisticGrowth).

Given such a library of domain knowledge, the task of learning process-based models takes two additional inputs, i.e., an incomplete model and measured data. Learning PBMs then proceeds in two phases: (1) instantiating the library of entity and process templates by using the incomplete model and (2) estimating the parameters in the resulting model structures to fit the measured data. Given the library of model fragments the former phase is formulated as a combinatorial search problem. Taking the incomplete model into account, one can instantiate the template entities and processes from the library into a set of specific components (entities and processes) to be included in the process-based model. The incomplete model represents modeling assumptions in terms of expected logical structure of the model, which limit the search space of the possible model structures, i.e., the combinations of model structures. Some of the combinations can be rejected as implausible, due to their inconsistency with the incomplete model in terms of presence or absence of certain processes. For example, an incomplete model of Predator-Prey dynamics can be obtained by removing the specific ExponentialGrowth and UnsaturatedPP processes and keeping their respective general templates. Together with the library presented in Table 1, this incomplete model results in 4 candidate model structures.

The latter task, i.e., estimating the model's parameters is formulated as an optimization task. Each of the candidate model structures considered during the search task is compiled into a system of equations, for which a parameter estimation task is solved to obtain values of the model parameters that best fit the observed data. The objective function usually considered for such problems is minimizing the discrepancy between the model simulation and the observed system behavior.

The basic inductive process-based modeling algorithm, performs exhaustive search through a constrained space of candidate process-based models, limiting the number of processes in the model [14]. More advanced approaches, such as Lagramge2.0 [15] and HIPM [19], allow for more sophisticated hierarchical constraints on the allowed process combinations. The most recent PBM tool, the ProBMoT [2], allows for complete modeling, parameter estimation, and simulation of process-based models.

The first input to ProBMoT is a library of domain knowledge. The next input consists of the modeling assumptions formalized as an incomplete model of the observed system. The third and final input is a set of measured data. Based on the incomplete model and the library of domain knowledge, ProBMoT generates a set of model structures. For each of these structures, parameter estimation is performed so that they best fit the measurements. The parameter estimation process is based on the meta-heuristic optimization framework jMetal 4.5 [20] that implements a number of global optimization algorithms. For this purpose, ProBMoT implements a variety of error functions such as root mean squared error (RMSE), root relative squared error (RRSE) and weighted root mean squared error (WRMSE). Parameter estimation relies on model simulation, for which ProBMoT employs the CVODE ODE solver from the SUNDIALS suite [21].

Finally, the output of ProBMoT is a set of complete models sorted according to their performance, i.e., the difference error between the simulation and the measured data. In this study, we use ProBMoT as the learning algorithm for inducing process-based models, i.e., learning the constituents of the process-based ensemble models, which will be introduced in the next section.

Ensembles of process-based models

Ensembles are an established method for improving the predictive performance of models in machine learning [5, 22]. An ensemble is a set of models (referred to as base models or ensemble constituents), that is expected to lead to a predictive performance gain over an individual model. In principle, any set of predictive models can be considered as an ensemble.

Once we have a set of predictive models, the question then arises as to how the individual predictions are to be combined into a single prediction. In traditional machine learning, this problem is tackled by using different combining schemes depending on the type of the base-models being aggregated. In the case of classification models that predict qualitative values, most often the predictions are combined by using different voting schemes.

The output of regression models that predict a single numeric value for a given input, can be combined by using the aggregation functions of average, weighted average and weighted median [23]. In this context, the output of an ensemble of process-based models resembles the one of time-series regression ensembles. However, in contrast to classical regression ensembles, the result of simulating an ensemble of process-based models is a whole trajectory instead of a single numeric value. Obtaining such a trajectory implies simulating each of the base models. This requires an initial value of the exogenous (state) variables and the trajectory of the exogenous (forcing) variables, which are then combined into an ensemble prediction. The different model simulations are combined time-point-wise: The combination is performed by well known methods usually applied in the case of numeric values, such as averaging, median, weighted average and weighted median [9].

When learning ensembles, the other important question is how to learn the constituent models of the ensemble. Methods for learning ensembles implicitly aim for diversity when learning the set of constituent models before aggregating their predictions. Based on how this diversity is achieved, we can distinguish two types of ensembles: heterogeneous and homogeneous. For learning heterogeneous ensembles, each of the base models is learned using a different learning algorithm (stacking [24]). On the other hand, in homogeneous ensembles, the individual base models are learned with the same learning algorithm, but from different samples of the training data, where the sampling variants include: sampling of the data instances (bagging [25], boosting [26]), sampling of the data features/attributes (random subspaces [10]) or both (bagging random subspaces [27], random forests [28]).

It has been theoretically and empirically shown that homogeneous ensembles, such as bagging and boosting, perform well for classification and regression problems [5, 28–30]. However, these methods, that sample data instances, can often be ineffective when the training data is relatively homogeneous. Additionally, when the dimensionality of the data (the feature space) is very high, learning such ensembles can be very ineffective and computationally complex.

The base level learning algorithm in this study is ProBMoT, and here we aim at learning homogeneous ensembles of process-based models. In the realm of process-based models, ensembles learned by sampling data instance, i.e., bagging and boosting, have demonstrated improved performance over individual models [8, 9], at the cost of increased computational complexity. In the continuation of this section, we outline the methods for sampling data instances in the context of process-based modeling. We then shift our focus to a method for learning ensembles of process-based models via sampling data features, i.e., sampling domain knowledge.

Sampling data instances for learning ensembles of process-based models. Bagging (bootstrap aggregation) refers to an approach, developed by Breiman [25], for constructing ensembles via bootstrap sampling with aggregation. This is one of the first and simplest ensemble learning methods, where data instances are uniformly sampled with-replacements to generate random samples (bootstrap replicates) of the training data, consequently used to learn a set of ensemble constituents. The learned base models are then combined by averaging their output (in the case of regression) or by voting (in the case of classification). Bagging ensembles successfully overcome the over-fitting problem most often when constitute instable models, i.e.

models that can change dramatically even for small changes in the training data. However, they are not that accurate when are constructed with stable models.

More sophisticated ensemble methods, such as boosting, obtain ensemble predictions by combing "imperfect" predictions made by base models learned on different distributions of the training data. The most notable implementation of boosting is the AdaBoost algorithm, proposed by Freund and Schapire [31], originally developed for tackling classification tasks, and later adapted by Drucker [23] for combining regressors. Like bagging, AdaBoost also re-samples the training data: However, instead of treating all instances equally (as in bagging), it prioritizes the more informative ones (i.e., those where large errors are currently made) for each subsequent iteration. Even though AdaBoost is very successful in tackling the over-fitting problem on a variety of tasks, its performance is prone to noise in the training data which can lead to lower performance as compared to single models and to other ensemble methods such as bagging [5].

For the task of bagging and boosting ensembles of process-based models, the candidate base models are learned from different samples of the measured data. The notable difference from bagging and boosting in the context of regression is that, in our case, the data instances have a temporal ordering, that has to be retained in each sample of the data. A detailed specification of how bagging and boosting are implemented in the context of process-based modeling can be found in the study by Simidjievski et al. [9].

Sampling domain knowledge for learning ensembles of process-based models. The random subspace method (RSM) is a homogeneous ensemble method developed by Ho [10], which constructs different variants of the training data by sampling the feature space. Each ensemble constituent is learned on all data instances and a subspace of the original feature space. The predictions of the learned base models are then combined via standard combining schemes for classification and regression, i.e., voting schemes and averaging techniques, respectively. The RSM has been reported to perform well for problems where the data dimensionality is very high or when there is a certain redundancy in the feature space [32].

In the context of learning ensembles of process based models, we can think of the feature space as being defined by the model components instantiated from the process templates. This space of components is determined by the number of process alternatives defined in the library of domain knowledge. Therefore, generating random samples of the feature space used in the traditional RSM, is analogous to generating random samples of the library of domain knowledge. The approach presented in this paper learns ensemble constituents from the whole data set using samples of the knowledge library. This is in contrast with bagging and boosting, where ensemble constituents are being learned on data samples using the same knowledge library or feature space.

The procedure for learning ensembles of process-based models via library sampling is presented in Algorithm 1. The procedure LS() takes five inputs: a library of domain knowledge (*lib*), a dataset consisting of training data (D_T) and validation data (D_V), an incomplete model (*incompleteModel*), a boolean variable (*allowDuplicates*), and an integer k denoting how many base models are to be generated. The output is a set of process-based models denoted with *Ensemble*.

Algorithm 1 Learn ensembles of process-based models via library sampling

```
1: procedure LS(lib, {$D_T$, $D_V$}, incompleteModel, allowDuplicates, k)
2: returns Ensemble
3:     Ensemble ← Ø          ▷ set of base models
4:     do
5:       $lib_S$ ← SAMPLE(lib)        ▷ randomly sample the library lib
6:       $modelList_i$ ← PROBMOT($lib_S$, $D_T$, incompleteModel)
7:       $bestModel_i$ ← RANK($modelList_i$, $D_V$)
```

```
8:       $\beta_i \leftarrow$ CONFIDENCE($bestModel_i$, $D_V$)      ▷ presented in Algorithm 2
9:       if $allowDuplicates$ then
10:          $Ensemble \leftarrow Ensemble \bigcup (bestModel_i, \beta_i)$
11:        else if $bestModel_i \notin Ensemble$ then
12:          $Ensemble \leftarrow Ensemble \bigcup (bestModel_i, \beta_i)$
13:        end if
14:    While SIZE($Ensemble$) $\neq k$
15: end procedure
```

Algorithm 2 Calculating confidence

```
1: function confidence($model$, $D$) returns $\beta$
2:     let $\hat{y}$            ▷ simulated system variable $y$
3:     let $y$            ▷ measured system variable $y$
4:     $\hat{y} \leftarrow$ SIMULATE($model$, $D$)
5:     $maxDisc \leftarrow \sup(|y_t - \hat{y}_t|)^2$         ▷ calculate max discrepancy between measurements $y$ and simulation $\hat{y}$, where $t = 0..N$ and $N$ is number of time-points in D
6:     $\bar{L} \leftarrow \sum_{t=0}^{N} \frac{|y_t - \hat{y}_t|^2}{maxDisc}$            ▷ calculate average loss
7:     $\beta \leftarrow \frac{\bar{L}}{1 - \bar{L}}$            ▷ calculate confidence
8: end function
```

For the task of sampling the library (SAMPLE(*lib*), line 5 in Algorithm 1) the process alternatives are randomly sampled (included or excluded) from the original library. The sampling algorithm takes as input the complete library and considers all the process templates defining more than one modeling choice. In turn, for each process template considered, it takes a random sample of the available modeling choices to be included in the sampled library. Note that the library sampling does not assume a uniform distribution of samples: the probability of a library sample is proportional to the size of the induced space of candidate models. In particular, the probability of a library sample lib_S of the whole library *lib* equals

$$P(lib_S) = \frac{|L_S|}{\sum\limits_{L_i \in \mathbb{P}(L)} |L_i|},$$

where L and $L_S \subseteq L$ correspond to the sets of candidate models induced by *lib* and lib_S (for a given incomplete model specification), respectively. Moreover, $|\cdot|$ denotes set cardinality and $\mathbb{P}(L)$ denotes the powerset of L, i.e., the set of all the possible subsets of L. For example, there are nine samples of the library from Table 1: one that generates four candidate models, four that induce two candidate models each, and four resulting in one candidate model each. The last four library samples are less likely selected (the probability of selecting each is 1/16) than the other five samples (1/4 for the first one, and 1/8 for each of the remaining four).

The process-based modeling algorithm PROBMOT(), (line 6 in Algorithm 1), takes as input the sample of the library of domain knowledge lib_S, time-series measurements of the observed dynamic system D_T, and an *incompleteModel* representing the modeling assumptions made by the modeler. The output of PROBMOT() is a list of process-based models, which is afterwards sorted according to their performance (line 7 in Algorithm 1). The output of the function RANK() (line 7 in Algorithm 1), i.e., the highest ranked model from each modeling task *i* (out of *k*) denoted as $bestModel_i$, becomes an ensemble constituent in the output *Ensemble*. The ranking can be based on the performance on a separate validation data set D_V or on the training sample (if $D_V = = D_T$).

Each ensemble constituent is paired with its own confidence β. The calculation of the CONFIDENCE() (Algorithm 2) function takes 2 inputs: the highest ranked model returned by ProBMoT and a data set D. First, the *model* is simulated on the data set D resulting in a trajectory $\hat{y}$. Based on the error at each time point in the trajectory, an average loss $\bar{L}$ is calculated for the *model* (line 6 in Algorithm 2). From this loss, a confidence measure β is calculated, where low values of β denote high confidence. The β coefficient is an indicator of the performance of the base model and is used in the process of simulating the ensemble, i.e., combining the simulations of the constituent models into an overall ensemble prediction when weighted combining schemes are considered (i.e., Weighted Average and Weighted Median).

Given the fact that the method always takes as input the same original library, there is a high probability of learning and choosing identical models from different library samples, thus filling the ensemble constituent set with multiple copies of the same model. To account for this, the method incorporates two different alternatives for generating the ensemble constituent set, i.e., with and without duplicates. For the former, referred to as *Library Sampling with Duplicates*, duplicates are allowed in the constituent set (line 9 in Algorithm 1). k library samples are generated (with k denoting number of ensemble iterations), and the best model out of each modeling task is chosen to be an ensemble constituent, regardless of whether that particular model was already in the constituent set or not. For the latter, referred to as *Library Sampling without Duplicates*, to incorporate more diversity in the ensemble, the method generates library samples (and performs modeling tasks) until the resulting ensemble contains k distinct constituents (line 11 in Algorithm 1).

Experimental Setup

In this section, we present the setup of the experiments used to evaluate the predictive performance of the ensembles of process-based models. We first introduce the data sets to be used in the experiments, then we briefly describe the two other ProBMoT inputs, the library of domain knowledge and the modeling assumptions. Next, we give an overview of the parameters of the algorithm for learning ensembles with bagging and boosting, used in the last set of experiments. Finally, we finish the section with the metrics used to measure the performance of process-based models and ensembles thereof.

The data. The performance of the proposed method for learning ensemble is evaluated on several tasks of modeling population dynamics in three aquatic ecosystems: Lake Bled in Slovenia, Lake Zurich in Switzerland and Lake Kasumigaura in Japan.

Lake Bled is located in the Julian Alps in north-western Slovenia and occupies an area of 1.4 km^2, with a volume of 0.0257 km^3, a maximum depth of 30.1 m and an average depth of 17.9 m. The measurements, performed by the Slovenian Environment Agency, consist of physical, chemical and biological data for the period from 1996 to 2002. All the measurements were performed once a month and depth-averaged for the upper 10 m of the lake.

Lake Zurich is located in the south-western part of the canton of Zurich in Switzerland. It has an average depth of 49 m, a volume of 3.9 km^3 and a surface area of 88.66 km^2. The data comprise measurements, performed by the Water Supply Authority of Zurich in the period from 1996 to 2002. They include profiles of physical, chemical and biological variables from 19 different sites, weight averaged to the respective epilimnion (top 10 m) and hypilimnion (bottom 10 m) depths. To obtain daily approximations, the data for both lakes were interpolated with a cubic spline algorithm and daily samples were taken from the interpolation [4, 33].

Lake Kasumigaura, located 60 km to the north-east of Tokyo, Japan, has an average depth of 4 m, a volume of 0.848km^3, and a surface area of 220km^2. The dataset comprises monthly measurements, taken in the period from 1986 to 1992. Similarly, to obtain daily approximations,

Table 2. Comparison of the predictive performance of a single model with that of ensembles of PBMs learned with library sampling, bagging and boosting.

Case	SingleModel	Library Sampling	Bagging	Boosting
B1	5.184 (4)	1.103 (3)	1.073 (2)	**1.065** (1)
B2	**0.938** (1)	1.143 (4)	1.085 (3)	0.947 (2)
B3	1.042 (4)	**0.794** (1)	0.968 (3)	0.941 (2)
B4	2.840 (4)	**0.750** (1)	0.760 (3)	0.791 (2)
B5	0.842 (2)	0.858 (3)	0.859 (4)	**0.698** (1)
K1	7.730 (4)	0.756 (2)	**0.736** (1)	0.925 (3)
K2	328.138 (4)	**0.878** (1)	1.429 (2)	1.642 (3)
K3	0.932 (4)	**0.827** (1)	0.867 (2)	0.923 (3)
K4	0.777 (3)	0.800 (4)	0.772 (2)	**0.705** (1)
K5	0.792 (4)	**0.732** (1)	0.734 (2)	0.781 (3)
Z1	**0.744** (1)	0.797 (3)	0.777 (2)	0.890 (4)
Z2	1.323 (4)	0.948 (3)	**0.883** (1)	0.893 (2)
Z3	29.463 (4)	**0.881** (1)	0.934 (2)	1.001 (3)
Z4	27.593 (4)	1.011 (3)	**0.964** (1)	0.982 (2)
Z5	1.489 (4)	1.130 (2)	**1.113** (1)	1.403 (3)

doi:10.1371/journal.pone.0153507.t002

the measurements were interpolated using linear interpolation and daily samples were taken from the interpolation [34].

In this paper, the same structure of a population dynamics model is used in all three aquatic ecosystems. It includes a single equation (ODE) for a system variable representing the phytoplankton biomass (measured as *chlorophyll-a* in Kasumigaura). The exogenous variables include the concentration of the zooplankton *Daphnia hyalina* (where available, i.e., for Bled and Zurich only), dissolved inorganic nutrients of nitrogen, phosphorus, and silica (measured as *ammonia* in Kasumigaura), as well as two input variables representing the environmental influence of water temperature and global solar radiation (light).

In the experiments, we use fifteen data sets, which are subsets of the above-mentioned measured data from the aquatic ecosystems in the three lakes. For each aquatic ecosystem, the original data set is split into seven single-year data sets. Five of these are used (one at a time) for training the ensemble constituents. From the remaining two, one is used for validating the models in the process of selecting the ensemble constituents, and one to measure the predictive performance of the learned process-based models and ensembles thereof. Therefore, fifteen learning experiments are performed; in each, we take a single-year training data set, learn a model using the train and the validation data sets, and test the predictive performance of the learned model on the test data set. In Table 2, which reports the experimental results, the experiments are labeled with the labels B1–B5, K1–K5 and Z1–Z5 corresponding to the training data set using in the experiment, where, e.g., K3 denotes the Lake Kasumigaura data set for the third year (i.e., 1988).

Note that, we follow the traditional experimental setup used in ecological modeling, where predictive models for a particular aquatic ecosystem are learned and tested on data from that same ecosystem. This is due to the fact that many environmental variables, corresponding to overall properties of the observed system (such as depth, volume and terrain configuration), are assumed to be constant in the modeling process. Additionally, some of the entities (variables) and ecological processes typically differ between ecosystems. Thus, models learned under this assumption can not be directly used in the context of other ecosystems.

The library of domain knowledge. In the performed experiments, we use the library of domain knowledge for modeling population dynamics in aquatic ecosystems, presented in čerepnalkoski et al. [2]. This library is based on the previous work of Atanasova et al. [35]. The library of domain knowledge, combined with the modeling assumptions, results in 18144 candidate models for Lake Kasumigaura and 27216 candidates for the other two lakes.

ProBMoT parameter settings. ProBMoT implements the Differential Evolution (DE) [36] method for parameter estimation. For the experiments performed in this paper, the DE parameters were set as follows: a population size of 50, strategy *rand/1/bin*, differential weight (*F*) and the crossover probability (*Cr*) of 0.6. The limit on the number of evaluations of the objective function is one thousand per parameter. For simulating the ODEs, the CVODE simulator is used with absolute and relative tolerances set to 10^{-3}.

Learning ensembles of process-based models with bagging and boosting. To properly assess the predictive performance of the proposed method for learning ensembles of process-based models, in our last set of experiments we compare it to the state-of-the-art methods for learning ensembles, i.e., bagging and boosting. Following the finding of Simidjievski et al. [9] the ensembles are learned with both bagging and boosting include 25 constituents, which are chosen based on their performance on a separate validation set and combined by averaging their predictions. These settings were chosen by following the same experimental setup that we use in this paper to select the appropriate design choices for learning ensembles with library sampling.

Note, however, an important difference between the setup of the bagging and boosting experiments and the experiments with library sampling. In the latter, we use the whole library of domain knowledge as described previously in subsection *The library of domain knowledge*. The use of this library is prohibitive for bagging and boosting, due to the high computational complexity considering the large space of candidate models in each learning iteration. To address this issue, we use a simplified version of the original library that results in 320 candidate model structures for Lake Kasumigaura and 128 candidates for the other two lakes. We have prepared the simplified library carefully, omitting only modeling alternatives (process templates) that are rarely observed to be among the top-ranked models in the single-model experiments with ProBMoT. The issue of computational complexity of learning ensembles of PBMs is further discussed in the subsection *The computational complexity of learning ensembles of PBMs* of the section *Results*.

Performance evaluation metrics. To evaluate the predictive performance of a given model *m*, we use the measure of relative root mean squared error (*ReRMSE*) [28], defined as:

$$ReRMSE(m) = \sqrt{\frac{\sum_{t=0}^{n} (y_t - \hat{y}_t)^2}{\sum_{t=0}^{n} (\bar{y} - \hat{y}_t)^2}}, \tag{1}$$

where n denotes the number of measurements in the test data set, y_t and $\hat{y}_t$ correspond to the *measured* and *predicted* (predictions are obtained by simulating the model m on the test set) value of the system variable y at time point t, and $\bar{y}$ denotes the mean value of y in the test data set. Note that the usual root mean squared error observed here, is relative to the standard deviation of the system variable in the test data, thus allowing us to compare the errors of models for different system variables with measured values on different scales (e.g., phytoplankton in different lakes).

Statistical comparison of performance. We observe and compare the predictive performance (in terms of *ReRMSE*) of the models learned using different algorithms on the 15 data sets. To properly assess the significance of the differences between the performances of models obtained with different algorithms, we follow the standard statistical procedure recommended by Demšar [37]. We use the corrected [38] Friedman test [39], followed by two post-hoc tests: the Nemenyi test [40] and the Bonferroni-Dunn test [41]. A positive outcome of the Friedman test indicates difference between the performances of the different algorithms considered. After the completion of the Fridman test, we proceed with performing post-hoc tests to identify which differences are statistically significant.

The first post-hoc test, i.e, the Nemenyi post-hoc test, computes the critical distance between the algorithm ranks at a given level of statistical significance (in this paper, the significance level threshold is set at 95%, $p = 0.05$). Only differences of the average ranks larger than the critical distance are considered significant; for those, we can claim that one algorithm outperforms (i.e., performs significantly better than) the other. This test is performed to obtain an assessment of the relative performance of the methods considered. In this paper, the Nemenyi post-hoc test is employed for comparison of different design decisions for the newly proposed method. The results of the Friedman-Nemenyi tests are depicted by average rank diagrams (as in Figs 2–6), where the critical distance is shown as a solid red line.

The second post-hoc test, i.e., the Bonferroni-Dunn post-hoc test, is performed to test how a proposed method performs in a comparison to other methods. This test is similar to the Nemenyi test, where a critical distance between the algorithm ranks is computed at a given level of significance (in this paper the significance level threshold is at 95%, $p = 0.05$), denotes how one method (i.e, an ensemble learned using the library sampling method) compares to the other existing methods for constructing ensembles of process-based models (i.e., bagging, boosting) and a single model, in terms of predictive performance. The results of the Friedman-Bonferroni-Dunn test is depicted by the average rank diagram (as in Fig 7), where the critical distance is shown as a dashed blue line.

Results

In this section, we present and discuss the results of the empirical evaluation. Given the fact that the method for learning ensembles with library sampling is novel, we first identify the most suitable design choices within the algorithm. We investigate what are the optimal choices of method for selecting the ensemble constituents, number of ensemble constituents, and combining method. The optimal choices are identified for both library sampling alternatives (with and without duplicates).

After making the design choices, we compare the predictive performance of the library-sampling ensembles with the performance of baseline ensembles, state-of-the-art bagging and boosting ensembles, and single models. The two baseline ensembles consist of the top-ten and ten randomly selected process-based models. Finally, we perform a comparative analysis of the computational complexities of different methods for learning ensembles of process-based models.

Design choices for the algorithm for learning library-sampling ensembles

The first design choice in an algorithm for learning ensembles of PBMs is related to the way of choosing the ensemble constituents. As previously outlined, the highest ranked model of each ensemble iteration is selected to be an ensemble constituent. The standard approach of ProBMoT to ranking candidate models is with respect to their performance on the training set. However, to avoid overfitting, these models can be re-ranked according to their performance on a separate validation data set. These experiments are performed using ensembles with 10, 25 and 50 constituents whose predictions are combined by averaging.

Fig 2 summarizes the performance comparison of the methods for choosing the base models to be included in the ensembles. The upper diagram (Fig 2A) depicts the results of the

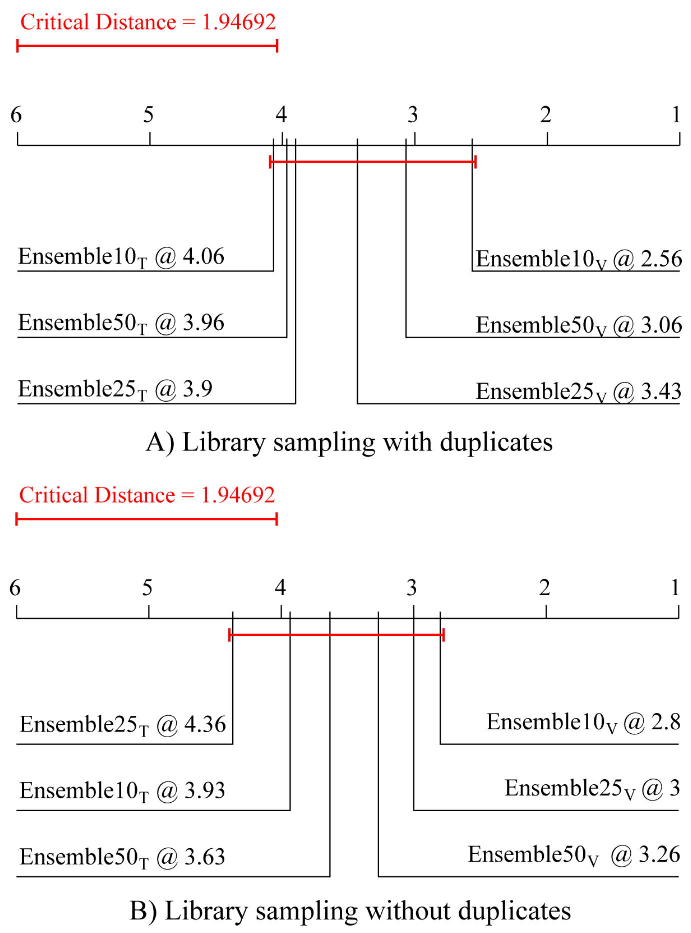


Fig 2. Comparison of the predictive performance in terms of average ranks of the two library sampling methods with different methods for choosing ensemble constituents. Average ranks of ensembles with 10, 25 and 50 constituents combined by averaging and selected differently (based on their train (subscript T) or validation (subscript V) performance). The average ranks refer to the predictive (testing) model performance averaged over the 15 experimental data sets, separately for the case of Library sampling (A) with and (B) without duplicates.

doi:10.1371/journal.pone.0153507.g002

Friedman-Nemenyi test for library sampling with duplicates, and the lower diagram (Fig 2B) for library sampling without duplicates. In both cases, choosing the base models based on their ranking on a separate validation set leads to better ensemble performance. However, in both cases, this improvement is not significant.

Earlier we conjectured that choosing ensemble constituents based on their performance on the training data might lead to overfitting. Fig 3 confirms this conjecture: In both cases (Fig 3A and 3B), ensembles comprising base models selected based on the performance on the training data exhibit significantly better descriptive/training performance than the ones selected based on their performance on a separate validation dataset. This is in-line with our previous findings

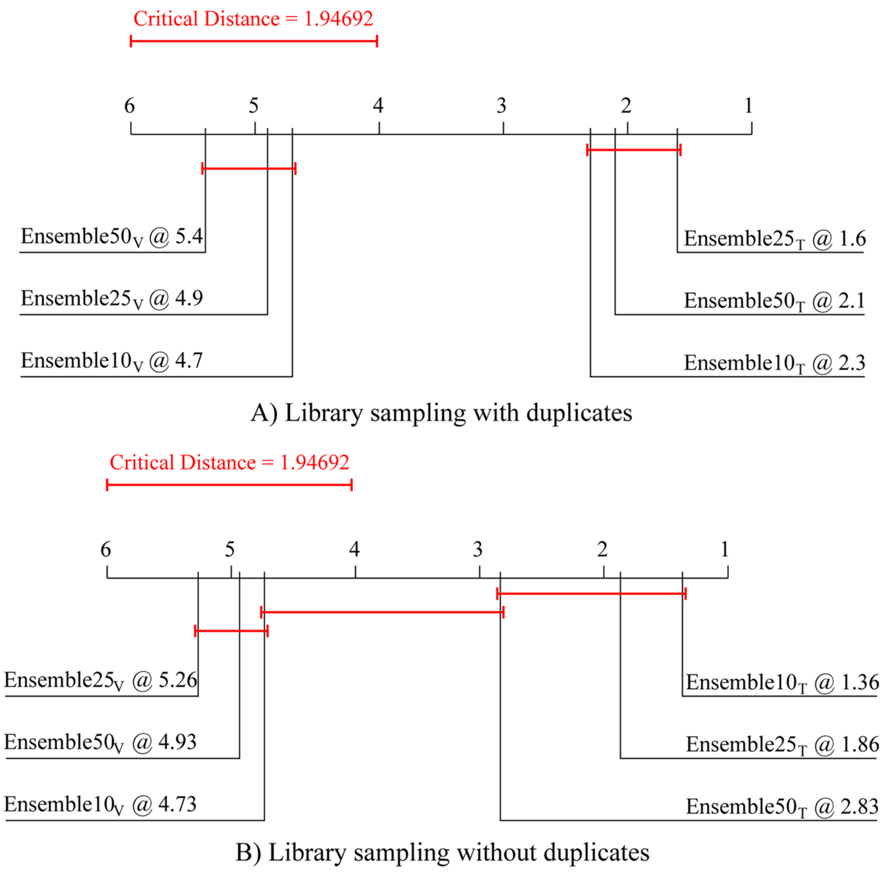
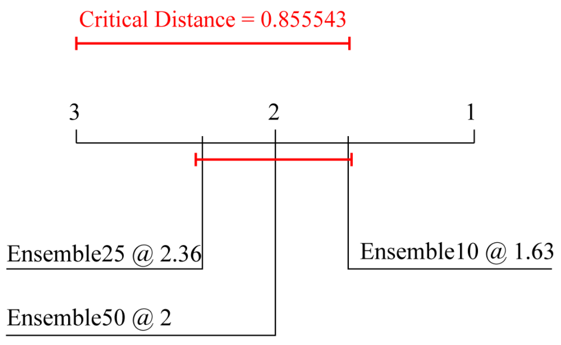


Fig 3. Comparison of the average ranks for descriptive performance (on training data) of the two library sampling methods with different methods for choosing ensemble constituents. Average ranks of ensembles with 10, 25 and 50 constituents combined by averaging and selected differently (based on their train (subscript T) or validation (subscript V) performance). The average ranks refer to the descriptive model (training) performance averaged over the 15 experimental data sets, separately for the case of Library sampling (A) with and (B) without duplicates.

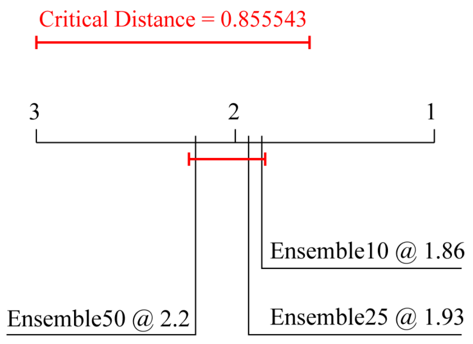
doi:10.1371/journal.pone.0153507.g003

for bagging and boosting [9], where the ensembles with constituents chosen on the bases of their training performance demonstrate a clear case of overfitting—while having significantly better performance on the training data these ensembles have worse predictive performance on unseen data.

Next, we focus on choosing the optimal number of base models in the ensembles of process-based models. To this end, we compare the predictive performance of ensembles consisting of 10, 25 and 50 base models, whose predictions are combined by averaging. The Friedman-Nemenyi diagram, presented in Fig 4, shows that the ensembles containing 10 ensemble constituents, for both types of ensembles (Fig 4A and 4B), lead to the best performance. Note however, that the observed difference in performance is not statistically significant.



A) Library sampling with duplicates



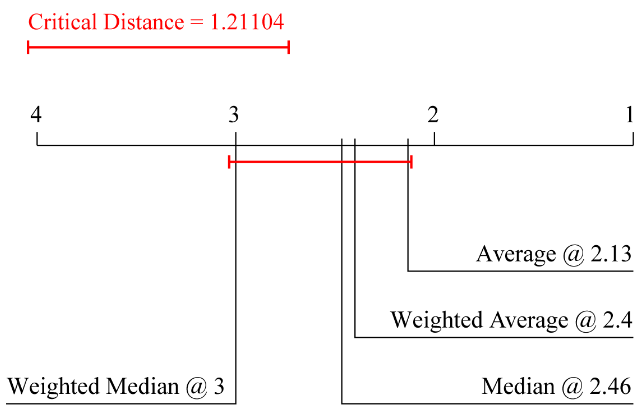
B) Library sampling without duplicates

Fig 4. Comparison of the average ranks for predictive performance of the two library sampling methods with different ensemble sizes. Average ranks of ensembles that include 10, 25, and 50 base models in terms of predictive performance averaged over the 15 experimental data sets for Library sampling (A) with and (B) without duplicates.

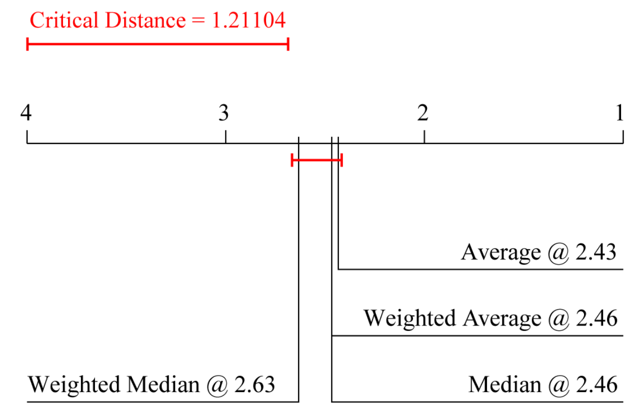
doi:10.1371/journal.pone.0153507.g004

Finally, we focus on choosing the most appropriate method for combining the simulations of the base models in the ensemble. We compare the performance aggregations of four methods that are often used in ensembles of regression models: the average, weighted average, median and weighted median methods. Fig 5 depicts the comparison of the average ranks for these methods both for library sampling with and without duplicates (Fig 5A and 5B, respectively), containing 10 constituents. It can be observed that in both cases the simple average method for aggregation performs (not significantly) best.

In summary, based on the performed experiments we make the following design decisions related to learning ensembles of process-based models with library sampling. First, the ensemble constituents are chosen based on their performance on a separate validation set, as they exhibit better performance than the ones chosen based on their performance on the original training data set. Second, the ensembles should consist of relatively small number(10) of base



A) Library sampling with duplicates



B) Library sampling without duplicates

Fig 5. Comparison of the average ranks for predictive performance of the two library sampling methods with four combining methods. Average ranks of the four methods for combining the simulations of base models (average, weighted average, median and weighted median) in terms of predictive performance averaged over the 15 experimental data sets for Library sampling (A) with and (B) without duplicates.

doi:10.1371/journal.pone.0153507.g005

models whose predictions should be combined by using the simple average method. Although the latter conclusion is based on results which are not statistically significant, it can be justified by following the parsimony principle. In all further experiments, we used these algorithm settings for learning ensembles for library sampling with and without duplicates.

Predictive performance of the library-sampling ensembles

Here we focus on testing our central hypothesis that the ensembles learned with library sampling improve the predictive performance of process-based models. However, to properly assess the performance of such ensembles, and whether/how it is related to the sampling of the library, we first compare the performance of both alternatives of library sampling to two

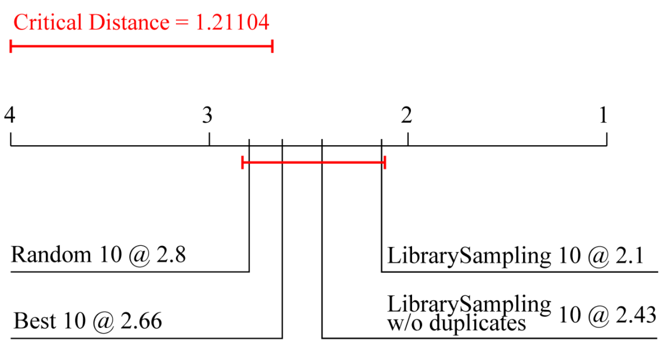


Fig 6. Comparison of the average ranks for predictive performance of the two library sampling methods to two alternative ensemble approaches. Average ranks of four different types of ensembles with 10 constituents combined by averaging. The constituents are: constructed via library sampling with and without duplicates, the 10 best models learned from the complete library, and 10 random models learned from the complete library. The average ranks of predictive performance are computed over the 15 data sets.

doi:10.1371/journal.pone.0153507.g006

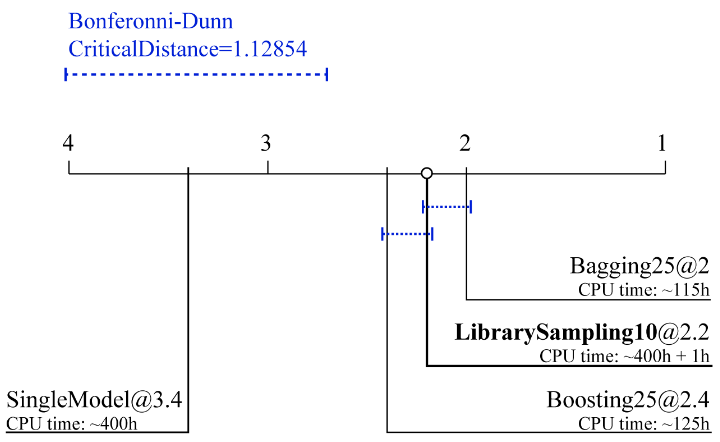


Fig 7. Comparison of the average ranks for predictive performance of the library sampling method to learning single models, bagging and boosting. Average ranks of the ensembles constructed by library sampling with 10 constituents combined by averaging to the performance of: a single model and two types of ensembles combined by averaging constructed by bagging and boosting (with 25 constituents). The ranks of the models in terms of their predictive performance and the average CPU times for learning them are averaged over the 15 data sets.

doi:10.1371/journal.pone.0153507.g007

baseline types of ensembles. The first baseline ensemble is comprised of the 10 best performing models learned using the complete library, combined by averaging. We refer to this ensemble as *Best10*. The second baseline ensemble is comprised of 10 randomly chosen models, also learned on the complete library and combined by averaging. We refer to this ensemble as *Random10*.

Fig 6 depicts the Friedman-Nemenyi comparison of the average ranks of the two library-sampling ensembles and the two baseline ensembles. It can be seen that the ensemble with library sampling with duplicates performs best, followed by the one which uses library sampling without duplicates. Note, however, that the test shows no significant difference in performance among the four approaches.

Given that the experiments performed so far did not show any substantial difference in the performance between the two alternatives for library sampling, we further investigated their structure. We found that the constituent set with 10 elements of the ensemble learned by library sampling with duplicates has on average 85% unique base models. Moreover, the method which samples the library without duplicates required on average 4 more iterations (per dataset) for learning an ensemble with 10 constituents. Given the results in Fig 6, and this insight into the structure of the ensembles, we follow the parsimony principle once again, and consider the method of library sampling with duplicates to be the more efficient and effective alternative when learning such ensembles.

In our last set of experiments, we assess the predictive performance of the method for learning ensembles via library sampling with the optimal design choices determined above. We compare its performance to the performance of a single model learned on the complete library, and ensembles learned using a reduced version of the library by the two state-of-the-art methods in process-based modeling, i.e., bagging and boosting with 25 constituents each and combined by averaging.

Fig 7 depicts the Friedman-Bonferroni-Dunn comparisons of the average ranks across the fifteen data sets. The results of the test show that the proposed ensembles with library sampling *significantly* outperform single process-based models. This result supports the central hypothesis of our paper that such ensembles improve the predictive performance over single process-based models. Note, however, that their predictive performance is slightly (not significantly) worse than bagging ensembles, and slightly (not significantly) better than boosting ensembles.

Fig 7 also presents the CPU time needed for learning all the ensembles and the single model. On average, learning ensembles with the library sampling method takes as much time as learning a single process-based model ($\approx$400 *h*). Note, however, that even though the remaining two methods of bagging and boosting take far less time than the previous methods, ($\approx$115 *h* and $\approx$125 *h*, respectively), they are learned from a reduced library. After a further investigation, which involved learning a single model on the smaller library (that took 4.5 to 5*h*), we estimate that it would take $\approx$9200 *h* and $\approx$10000 *h*, for bagging and boosting of process-based models using the original libraries, respectively. This would result in substantially (by a factor of $\approx$25) worse computational efficiency as compared to learning library-sampling ensembles.

So far, our comparison of the methods focused on their average ranks, which do not reveal the actual performance of the models obtained with the different methods. Table 2 reports these performances of the single process-based models and the three ensembles of PBMs learned with library sampling, bagging and boosting. Note that the results reported in the table confirm the superiority of ensembles to a single model. The ensemble methods are far more robust than single models, which severely under-perform in 6 out of the 15 cases: in these 6 cases, the single models have a ReRMSE of over 2. In the two cases where the single models outperforms the ensembles (B2 and Z1), the difference in performance when compared with the top-ranked ensemble method for the particular case is minor. On the other hand, ensemble methods can not be differentiated in terms of robustness: all three methods for learning ensembles of PBMs are equally robust. The differences in performances among the models obtained with the different ensemble methods are minor and often negligible. The relative winner is the proposed algorithm for library sampling, that outperforms the other two methods in 6 out of 15 cases. In terms of predictive performance, the method of library sampling is competitive and even slightly better than the state-of-the-art ensemble methods of bagging and boosting PBMs, leading to more accurate predictive models.

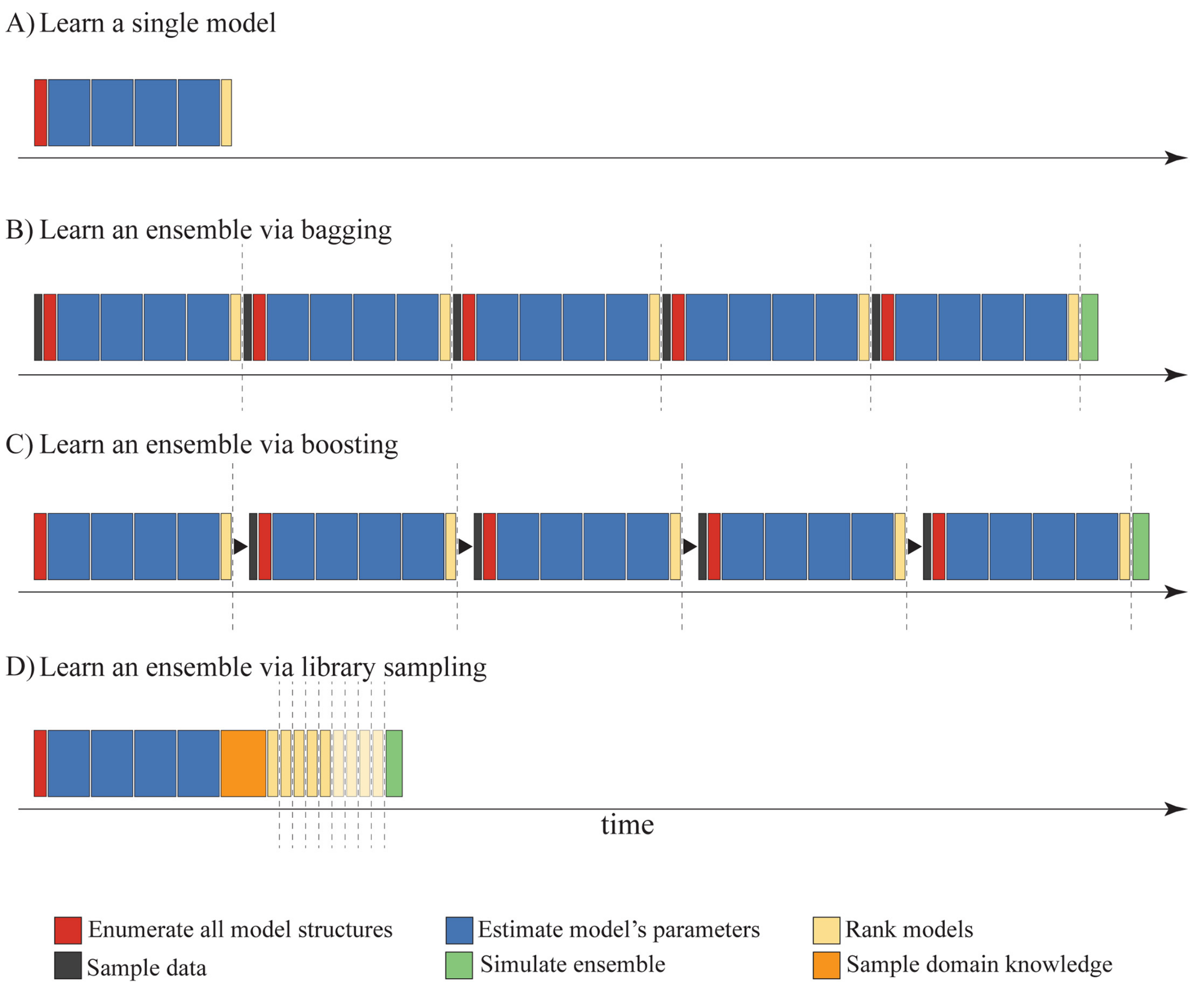

A) Learn a single model
B) Learn an ensemble via bagging
C) Learn an ensemble via boosting
D) Learn an ensemble via library sampling
time
Enumerate all model structures
Estimate model's parameters
Rank models
Sample data
Simulate ensemble
Sample domain knowledge

Fig 8. Comparison of the computational complexity of learning a single model and three methods for learning ensembles. A graphical representation of the time complexity of learning a (A) single process-based model and ensembles of process-based models with five constituents with (B) bagging, (C) boosting and (D) library sampling, using the library presented in Table 1.

doi:10.1371/journal.pone.0153507.g008

Computational complexity of learning ensembles of PBMs

Since the main focus of this study is the efficient learning of ensembles of process-based models, we first need to establish the complexity of learning a single model. Recall from Section *Process-based models*, that the algorithm for learning process-based models consists of two main sub-tasks: enumerating all possible model structures, and estimating the parameters of each of them. Fig 8A presents a diagram of relative execution times for each of the tasks through the prism of learning an example Predator-Prey model as described in Section *Process-based models*. The first task, structure enumeration (red box), uses a traversal algorithm through the space of model components, which is linear in the resulting number candidate models N [42]. In this example, it results in 4 candidate models. Second, for each of these candidates, a parameter estimation task is performed (blue box), the efficiency of which is related to the number of parameters each candidate model has and the number of observed time points. Note that, even though the overall complexity of the parameter estimation task is $\mathcal{O}(N)$, in most cases more than 99% of the computational time is spent in this phase. At the end of each modeling task, the learned models are ranked (yellow box) based on their performance on the training/validation dataset. The ranking task is performed by an insertion sort algorithm, based on the models' performances and has a complexity of $\mathcal{O}(NlogN)$ (given that a sorted list of models is maintained after every model construction), where N is the number of candidate models. For this paper we define the complexity of one iteration of ProBMoT, i.e. obtaining one process-based model as the benchmark unit for assessing the complexity of learning different ensembles of process-based models.

For analyzing the computational complexity of the different methods for learning ensembles of process-based models, we demonstrate learning Predator-Prey ensemble models with five constituents. Fig 8B presents the time needed for learning an ensemble using the bagging method. It is essentially a repetition of the tasks needed for learning a single model for every ensemble constituent (dashed line), with the exception of two additional tasks: one for sampling the training data (gray box) at the beginning of each ensemble iteration, and one for simulating the ensemble at the end (green box). Even though Fig 8B depicts the serial implementation of bagging, this algorithm handles the processes of learning different base models completely independently. Thus, it can be parallelized to handle different tasks with different bootstrap replicates on different CPUs, which performance-wise is very useful for computationally intensive learning tasks such as process-based modeling. The complexity of learning an ensemble with boosting (Fig 8C) resembles the one of bagging. However, the boosting algorithm cannot be parallelized, as each new boosting iteration depends on the outcome of the previous one. This makes such ensembles the most inefficient to learn, which is most strongly felt when large libraries and many constituents are considered.

The last method that we investigate is learning ensembles by sampling the library of domain knowledge (Fig 8D). While the algorithm (as presented in Algorithm 1) is iterative, it can be implemented much more efficiently. Instead of sampling the space of components (i.e., sampling the domain knowledge) and running ProBMoT with each sample of the library, we can sample the generated search space and choose from the candidate models. First, all the models from the original library are generated, and their parameters are estimated accordingly. Next, we generate all the necessary library samples (orange box), and perform the task of searching and sorting models which are determined by the particular subsample of the library. By transforming the sampling problem from sampling domain knowledge to sampling the model structures from the complete search space, the number of ProBMoT runs is minimized (to 1), consequently substantially gaining computational efficiency, as compared to the other ensemble methods using ProBMoT, i.e., bagging and boosting. Earlier it was stated that selecting the constituent set for such ensembles can be performed in two ways: with and without duplicates. For the former, the execution time is correlated to the number of iterations needed (which for this example is 5). For the latter, the execution time depends on the random generator, and for this example it can take from a minimum 5 (yellow boxes) to a maximum 9 iterations (additional 4 opaque-yellow boxes). Finally, at the end, similarly to the previous approaches, an ensemble simulation task is performed.

Summary

The results of the experiments show the following.

- The optimal design choices for the algorithm for learning library-sampling ensembles of process-based models are as follows. First, it is better to select the ensemble constituents using a separate validation dataset that has not been used to learn them (Figs 2 and 3). Second, the optimal number of constituent models is small (Fig 4). Finally, the simple average is the optimal method for combining the simulations/predictions of the ensemble constituents (Fig 5). Note that these design choices are virtually identical to the ones made for the bagging and boosting ensembles of process-based models [9].
- Library-sampling ensembles outperform the two baseline ensembles consisting of top-ten and ten randomly selected models (Fig 6). Moreover, despite the additional computational effort, removing duplicates in the set of ensemble constituents does not improve their predictive performance. Most notably, library-sampling ensembles significantly outperform single models and have comparable predictive performance to bagging (insignificantly worse average rank and slightly higher number of wins), the best performing state-of-the-art ensemble method for process-based modeling (Fig 7 and Table 2).
- The comparative analysis of the computational performances reveals that library-sampling ensembles are learned in a time comparable to the time needed to learn a single model (Figs 7 and 8). This is orders of magnitude faster when compared to its bagging and boosting counterparts, where the time needed equals the number of ensemble constituents multiplied by the time needed to learn a single model. In particular, the speed-up factor equals the number of bagged/boosted process-based models.

In sum, the library-sampling ensembles represent an important advance over the state-of-the-art methods (bagging and boosting) for learning ensembles of process-based models: They provide orders-of-magnitude improvement in computational efficiency of ensembles without impairing their predictive performance.

Discussion

The machine learning literature provides various frameworks for explaining the performance improvement gained by using ensemble methods. While the results on the positive influence of ensemble constituents' diversity on the performance are inconsistent [43], there is a general consensus that the bias-variance decomposition of the predictive error allows for a plausible explanation. Ensembles tend to reduce the variance component of the predictive error, while not increasing the bias component at the same time [44]. This is what happens in the case of learning ensembles of process-based models: by averaging the predictions of several models learned on the same dataset, we reduce the variance component of the predictive error. Moreover, we conjecture that ensembles of process-based models also reduce their bias. This is due to the fact that averaged process-model simulations can lead to predictions that are out of the scope of a single process-based model. Therefore, ensembles of process-based models (and models of dynamic systems in general) have the potential to extend the original space of individual models leading to a reduction of the bias component of the predictive error. Validating this hypothesis is beyond the scope of this paper, but will be considered in further work.

This paper builds upon previous work on learning ensembles for modeling dynamic systems. More specifically, it extends the scope of learning ensembles of process-based models for long-term predictive tasks, i.e., bagging and boosting [9], by introducing a novel approach for constructing efficient ensembles with satisfactory predictive performance. Second, the methods presented in this paper provide accurate predictions of the unobserved future system behavior, in contrast to Bridewell et al. [11], who build ensembles that provide accurate description of the observed system behavior.

In a broader sense, the work on library-sampling ensembles extends the state-of-the art methods for process-based modeling of population dynamics in ecology [2, 3, 14]. However, while these studies successfully for modeled the observed behavior of real-world aquatic ecosystems, the prediction of future system behavior is out of their scope. In similar context, the studies of Whigham and Recknagel [45] and Cao et al. [46] discuss the predictive performance of process-based models in a lake ecosystem. However, they assume a given model structure and by employing genetic algorithms mainly focus on the task of parameter identification of different model structures.

The novel ensemble method proposed in this paper aims at improving the generalization power of the process-based modeling approach while having reasonable computational complexity for modeling tasks with relatively large candidate model space. However, when learning such ensembles, there is a trade-off between the predictive performance of the ensemble and its understandability. Here, the gain in predictive accuracy comes at the cost of losing the understandability of the learned ensembles. Nonetheless, methods for improving the comprehensibility of ensembles can be developed, similar to the methods for integrating ensemble constituents in a single model proposed by Bridewell et al [11].

Conclusion

In this paper, we address the task of learning ensembles of process-based models. We design, implement and evaluate a novel methodology for learning ensembles by sampling the library of domain knowledge. This methodology for learning ensembles of process-based models is the major contribution of our paper, since it improves the state-of-the-art of learning process-based models in two directions: efficiency and performance.

First, it improves the state-of-the art of learning ensembles of process-based models with a new, computationally efficient ensemble method based on library sampling. The computational efficiency of the method allows for applications to domains where a rich library of domain knowledge is available leading to a large space of candidate models. In such cases, the standard, iterative methods of bagging and boosting are not applicable due to the prohibitively high computational costs. To apply them, we need to handcraft the library of domain knowledge, omit modeling alternatives, and thus simplify the space of candidate model structures. In addition to being more efficient, the proposed method most often leads to models with better predictive performance as compared to the models obtained with the state-of-the-art ensemble methods of bagging and boosting.

Second, it improves the predictive performance of the constructed ensembles, which mainly contributes to the realm of ecological modeling. The results of the performed experimental evaluation confirm our central hypothesis that ensembles constructed via sampling the library of domain knowledge provide more accurate predictions of concentrations of species in an aquatic ecosystems than a single process-based model. This is a significant improvement of predictive performance over the state-of-the-art models of population dynamics, which, while focusing on providing an accurate explanation of the behavior of the observed system, struggle to achieve a satisfactory performance at predicting population dynamics over a long periods [4, 34]. Note that these results are consistent over experiments with data from three real-world aquatic ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan, and Lake Zurich in Switzerland.

While our comparative empirical study is limited to the domain of modeling population dynamics, the proposed approach to learning ensembles of process-based models is general enough to be applied to any other domain and to any other type of models of dynamic systems. To proceed with process-based modeling in other domains, one has to encode a library of domain-specific modeling knowledge. Such knowledge is available for various domains including epidemiology and modeling gene regulatory networks [47], net carbon production [48] and protein interactions [49, 50]. Some of these applications of process-based modeling show the applicability of the proposed approach to the very active fields of systems biology and bioinformatics that deal with numerous system identification problems. The efficiency and predictive performance of the library-sampling ensembles bears a promise for further applications, since combining predictive models into ensembles has been proved to work well for various computational biology problems [51–53].

We have identified several limitations of our approach that can be addressed in further work. First, note that the experiments performed in this paper are limited to modeling population dynamics in three lake ecosystems. We intend to investigate the generality of our approach and extend the scope of learning ensembles of process-based models of population dynamics to other aquatic environments, such as marine ecosystems [14] and river ecosystems [54]. Next, considering the understandability of these ensembles, we plan to follow ideas from [11] and improve our methodology by incorporating understandable structure into the resulting ensemble. Also, investigating the hypotheses about the source of the predictive performance improvement in terms of bias-variance decomposition of the predictive error, requires careful and extensive empirical analysis using the setup of Brown et al [44]. Finally, given the nature of the ensembles of process-based models learned with library sampling and bagging (i.e., sampling the feature space and sampling the data instances), and their superior performance over the individual process-based model in a predictive setting, we intend to combine these two methods. A study performed by Panov and Džeroski [27] indicates that combining two types of homogeneous ensembles, i.e. bagging and RSM, can lead to ensembles with even better predictive performance compared to each of the methods separately, while still being very efficient to construct.

Acknowledgments

We thank Nataša Atanasova for sharing her expertise on the systems of Lake Bled and Lake Kasumigaura, and Anne Dietzel for the data and expertise about Lake Zurich.

Author Contributions

Conceived and designed the experiments: NS LT SD. Performed the experiments: NS. Analyzed the data: NS LT. Wrote the paper: NS LT SD.

References

1. Taškova K, Šilc J, Atanasova N, Džeroski S. Parameter estimation in a nonlinear dynamic model of an aquatic ecosystem with meta-heuristic optimization. Ecological Modelling. 2012; 226:36–61. doi: 10.1016/j.ecolmodel.2011.11.029
2. Čerepnalkoski D, Taškova K, Todorovski L, Atanasova N, Džeroski S. The influence of parameter fitting methods on model structure selection in automated modeling of aquatic ecosystems. Ecological Modelling. 2012; 245(0):136–165. doi: 10.1016/j.ecolmodel.2012.06.001
3. Borrett SR, Bridewell W, Langley P, Arrigo KR. A method for representing and developing process models. Ecological Complexity. 2007; 4(1-2):1–12. doi: 10.1016/j.ecocom.2007.02.017
4. Atanasova N, Todorovski L, Džeroski S, Remec R, Recknagel F, Kompare B. Automated modelling of a food web in Lake Bled using measured data and a library of domain knowledge. Ecological Modelling. 2006; 194(1-3):37–48. doi: 10.1016/j.ecolmodel.2005.10.029

5. Dietterich TG. Ensemble Methods in Machine Learning. In: Proceedings of the 1st International Workshop on Multiple Classifier Systems. London, UK: Springer; 2000. p. 1–15.

6. Maclin R, Opitz D. Popular ensemble methods: An empirical study. Journal of Artificial Intelligence Research. 1999; 11:169–198.

7. Rokach L. Ensemble-based classifiers. Artificial Intelligence Review. 2010 Feb; 33(1-2):1–39. doi: 10.1007/s10462-009-9124-7

8. Simidjievski N, Todorovski L, Džeroski S. Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems. Ecological Modelling. 2015; 306:305–317. doi: 10.1016/j.ecolmodel.2014.08.019

9. Simidjievski N, Todorovski L, Džeroski S. Predicting long-term population dynamics with bagging and boosting of process-based models. Expert Systems with Applications. 2015; 42(22):8484–8496. doi: 10.1016/j.eswa.2015.07.004

10. Ho TK. The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998 Aug; 20(8):832–844. doi: 10.1109/34.709601

11. Bridewell W, Asadi NB, Langley P, Todorovski L. Reducing overfitting in process model induction. In: Proceedings of the 22nd International Conference on Machine Learning. ICML'05. New York, NY, USA: ACM; 2005. p. 81–88.

12. Langley PW, Simon HA, Bradshaw G, Zytkow JM. Scientific Discovery: Computational Explorations of the Creative Processes. Cambridge, MA, USA: The MIT Press; 1987.

13. Washio T, Motoda H, Niwa Y. Enhancing the Plausibility of Law Equation Discovery. In: Proceedings of the 17th International Conference on Machine Learning, ICML'00. San Francisco, CA, USA: Morgan Kaufmann; 2000. p. 1127–1134.

14. Bridewell W, Langley PW, Todorovski L, Džeroski S. Inductive Process Modeling. Machine Learning. 2008; 71:1–32. doi: 10.1007/s10994-007-5042-6

15. Todorovski L, Džeroski S. Integrating Domain Knowledge in Equation Discovery. In: Džeroski S, Todorovski L, editors. Computational Discovery of Scientific Knowledge. vol. 4660. Springer; 2007. p. 69–97.

16. Kocev D, Džeroski S. Habitat modeling with single- and multi-target trees and ensembles. Ecological Informatics. 2013; 18:79–92. doi: 10.1016/j.ecoinf.2013.06.003

17. Blockeel H, Raedt LD. Top-down induction of first-order logical decision trees. Artificial Intelligence. 1998; 101(1-2):285–297. doi: 10.1016/S0004-3702(98)00034-4

18. Knudby A, Brenning A, LeDrew E. New approaches to modelling fish-habitat relationships. Ecological Modelling. 2010 Feb; 221(3):503–511. doi: 10.1016/j.ecolmodel.2009.11.008

19. Todorovski L, Bridewell W, Shiran O, Langley PW. Inducing hierarchical process models in dynamic domains. In: Proceedings of the 20th National Conference on Artificial Intelligence, NAI'05. AAAI Press; 2005. p. 892–897.

20. Durillo JJ, Nebro AJ. jMetal: A Java framework for multi-objective optimization. Advances in Engineering Software. 2011; 42:760–771. doi: 10.1016/j.advengsoft.2011.05.014

21. Cohen SD, Hindmarsh AC. CVODE, a stiff/nonstiff ODE solver in C. Computers in Physics. 1996 Mar; 10(2):138–143. doi: 10.1063/1.4822377

22. Okun O, Valentini G, Re M. Ensembles in Machine Learning Applications. vol. 373. Springer; 2011.

23. Drucker H. Improving Regressors using Boosting Techniques. In: Proceedings of the 14th International Conference on Machine Learning, ICML'97. San Francisco, CA, USA: Morgan Kaufmann; 1997. p. 107–115.

24. Wolpert DH. Stacked Generalization. Neural Networks. 1992; 5:241–259. doi: 10.1016/S0893-6080(05)80023-1

25. Breiman L. Bagging predictors. Machine Learning. 1996a; 24(2):123–140. doi: 10.1023/A:1018054314350

26. Schapire RE. The boosting approach to machine learning: An overview. In: Denison DD, Hansen MH, Holmes CC, Mallick B, Yu B, editors. Nonlinear Estimation and Classification. LNCS. Springer; 2003. p. 149–173.

27. Panov P, Džeroski S. Combining Bagging and Random Subspaces to Create Better Ensembles. In: Proceeding of the 7th International Symposium on Advances in Intelligent Data Analysis, IDA'07. vol. 4723 of LNCS. Springer; 2007. p. 118–129.

28. Breiman L. Classification and Regression Trees. Chapman & Hall, London, UK; 1984.

29. Bauer E, Kohavi R. An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. Machine Learning. 1999; 36(1-2):105–139. doi: 10.1023/A:1007515423169

30. Freund Y, Schapire RE. A Short Introduction to Boosting. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence, IJCAI'99. Morgan Kaufmann; 1999. p. 1401–1406.

31. Freund Y, Schapire RE. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Computer and System Sciences. 1997; 55(1):119–139. doi: 10.1006/jcss.1997.1504

32. Ho T. Complexity of Classification Problems and Comparative Advantages of Combined Classifiers. In: 1st International Workshop on Multiple Classifier Systems. vol. 1857 of LNCS. Springer; 2000. p. 97–106.

33. Dietzel A, Mieleitner J, Kardaetz S, Reichert P. Effects of changes in the driving forces on water quality and plankton dynamics in three Swiss lakes—long-term simulations with BELAMO. Freshwater Biology. 2013; 58(1):10–35. doi: 10.1111/fwb.12031

34. Atanasova N, Recknagel F, Todorovski L, Džeroski S, Kompare B. Computational Assemblage of Ordinary Differential Equations for Chlorophyll-a Using a Lake Process Equation Library and Measured Data of Lake Kasumigaura. In: Recknagel F, editor. Ecological Informatics. Springer; 2006. p. 409–427.

35. Atanasova N, Todorovski L, Džeroski S, Kompare B. Constructing a library of domain knowledge for automated modelling of aquatic ecosystems. Ecological Modelling. 2006; 194(1-3):14–36. doi: 10.1016/j.ecolmodel.2005.10.002

36. Storn R, Price K. Differential Evolution—A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization. 1997; 11(4):341–359. doi: 10.1023/A:1008202821328

37. Demšar J. Statistical Comparisons of Classifiers over Multiple Data Sets. Journal of Machine Learning Research. 2006; 7:1–30.

38. Iman RL, Davenport JM. Approximations of the critical region of the Friedman statistic. Communications in Statistics—Theory and Methods. 1980; 9(6):571–595. doi: 10.1080/03610928008827904

39. Friedman M. A comparison of alternative tests of significance for the problem of m rankings. The Annals of Mathematical Statistics. 1940; 11(1):86–92. doi: 10.1214/aoms/1177731944

40. Nemenyi PB. Distribution-free Multiple Comparisons. Princeton University; 1963.

41. Dunn OJ. Multiple Comparisons Among Means. Journal of the American Statistical Association. 1961; 56(293):pp. 52–64. doi: 10.1080/01621459.1961.10482090

42. Čerepnalkoski D. Process-Based Models of Dynamical Systems: Representation and Induction. Jožef Stefan International Postgraduate School, Ljubljana; 2013.

43. Kuncheva LI, Whitaker C. Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy. Machine Learning. 2003; 51(2):181–207. doi: 10.1023/A:1022859003006

44. Brown G, Wyatt J, Harris R, Yao X. Diversity creation methods: A survey and categorisation. Journal of Information Fusion. 2005; 6(1):5–20. doi: 10.1016/j.inffus.2004.04.004

45. Whigham PA, Recknagel F. Predicting chlorophyll-a in freshwater lakes by hybridizing process-based models and genetic algorithms. Ecological Modelling. 2001; 146(1-3):243–251. doi: 10.1016/S0304-3800(01)00310-6

46. Cao H, Recknagel F, Cetin L, Zhang B. Process-based simulation library SALMO-OO for lake ecosystems. Part 2: Multi-objective parameter optimization by evolutionary algorithms. Ecological Informatics. 2008; 3(2):181–190. doi: 10.1016/j.ecoinf.2008.02.001

47. Tanevski J, Todorovski L, Džeroski S. Learning stochastic process-based models of dynamical systems from knowledge and data. BMC Systems Biology. 2016; 10(1):30. doi: 10.1186/s12918-016-0273-4 PMID: 27005698

48. Todorovski L, DÅ¾eroski S, Langley P, Potter C. Using equation discovery to revise an Earth ecosystem model of the carbon net production. Ecological Modelling. 2003; 170(2-3):141–154. doi: 10.1016/S0304-3800(03)00222-9

49. Taškova K, Korošec P, Šilc J, Todorovski L, Džeroski S. Parameter estimation with bio-inspired meta-heuristic optimization: modeling the dynamics of endocytosis. BMC Systems Biology. 2011; 5(1):1–26.

50. Tanevski J, Todorovski L, Kalaidzidis Y, Džeroski S. Domain-specific model selection for structural identification of the Rab5-Rab7 dynamics in endocytosis. BMC Systems Biology. 2015; 9(1):1–17. doi: 10.1186/s12918-015-0175-x

51. Liu B, Fang L, Long R, Lan X, Chou KC. iEnhancer-2L: a two-layer predictor for identifying enhancers and their strength by pseudo k-tuple nucleotide composition. Bioinformatics. 2016; 32(3):362–369. doi: 10.1093/bioinformatics/btv604 PMID: 26476782

52. Chen J, Wang X, Liu B. iMiRNA-SSF: Improving the Identification of MicroRNA Precursors by Combining Negative Sets with Different Distributions. Scientific Reports. 2016; 6:19062. doi: 10.1038/srep19062 PMID: 26753561

53. Ackermann M, Clément-Ziza M, Michaelson JJ, Beyer A. Teamwork: Improved eQTL Mapping Using Combinations of Machine Learning Methods. PLoS ONE. 2012; 7(7):1–8. doi: 10.1371/journal.pone.0040916

54. Škerjanec M, Atanasova N, Čerepnalkoski D, Džeroski S, Kompare B. Development of a knowledge library for automated watershed modeling. Environmental Modelling & Software. 2014; 54:60–72. doi: 10.1016/j.envsoft.2013.12.017

Chapter 7

Learning Ensembles via Sampling Both Data Instances and the Library of Domain Knowledge

In this chapter, we present the results of the empirical evaluation of the bagging of random library subsamples (BRLS) method for learning ensembles of process-based models. In particular, we first identify the most suitable design choices for the method. These include the approach to selecting the ensemble constituents, the number of ensemble constituents, and how they should be combined for optimal performance. Next, we analyze the predictive performance of such ensembles of process-based models in terms of whether they outperform individual process-based models. In addition, we compare their performance to ensembles of process-based models learned with bagging, boosting and random library subsampling.

7.1 Design Choices

The first design choice in an algorithm for learning ensembles of PBMs is related to the way of choosing the ensemble constituents. As previously outlined, the highest ranked model of each ensemble iteration is selected to be an ensemble constituent. The standard approach of ProBMoT to ranking candidate models is to rank them with respect to their performance on the training set. However, to avoid overfitting, these models can be re-ranked according to their performance on a separate validation data set. These experiments are performed using ensembles with 10, 25 and 50 constituents whose predictions are combined by averaging.

Table 7.1 and Figure 7.1 summarize the results of the comparison of the methods for choosing the base models to be included in the ensembles. From the table, we can see that for all but three data sets (B1, B3 and B5), the selection method based on a separate validation data set outperforms the regular method based on the training data. The Friedman-Nemenyi test (and the corresponding diagram in Figure 7.1) confirms the observed superiority of the validation method: All methods that use validation are ranked better than those that use the training set. However, this improvement is not statistically significant.

Note that we made the implicit conjecture that choosing ensemble constituents based on their performance on the training data might lead to overfitting. The results presented in Table 7.2 and Figure 7.2 confirm the validity of this conjecture. From the table, we can see that the method for selecting constituents based on the training data consistently leads to significantly lower errors on the training data: ensembles using selection based on

Table 7.1: Comparison of the predictive performance (RRMSE on the testing set) of the two methods for choosing ensemble constituents (V=validation, T=training).

Case	Ens.10_V	Ens.25_V	Ens.50_V	Ens.10_T	Ens.25_T	Ens.50_T
B1	1.113	1.102	1.128	**1.045**	1.071	1.077
B2	1.238	1.237	**1.214**	1.248	1.268	1.259
B3	0.912	0.946	0.898	**0.706**	0.750	0.766
B4	0.811	**0.767**	0.804	0.809	0.737	0.757
B5	0.628	0.627	0.667	**0.579**	0.601	0.618
K1	0.749	0.639	**0.634**	1.484	1.310	1.308
K2	0.860	**0.830**	0.890	0.874	0.955	1.090
K3	**0.701**	0.735	0.733	NaN	0.954	0.859
K4	0.768	0.764	**0.751**	0.782	0.746	0.771
K5	1.004	0.889	**0.851**	0.781	0.674	0.668
Z1	**1.004**	1.461	1.218	1.130	1.087	1.133
Z2	**0.825**	0.844	0.885	1.074	0.946	1.234
Z3	0.934	0.901	**0.897**	1.611	1.183	1.073
Z4	1.034	**1.005**	1.007	1.054	1.074	1.058
Z5	1.009	0.999	**0.999**	1.389	1.271	1.216

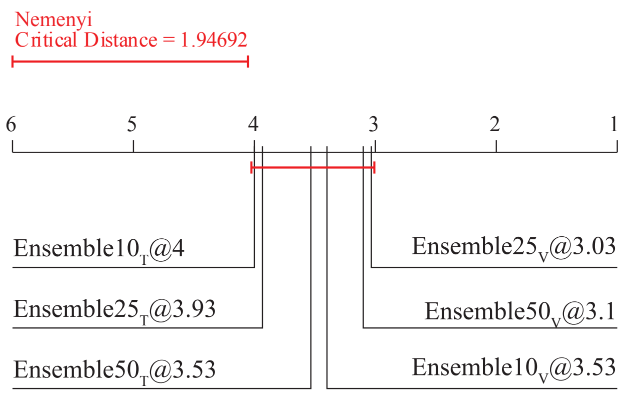


Figure 7.1: Average ranks of ensembles with 10, 25 and 50 constituents combined by averaging and selected differently (based on their train (subscript T) or validation (subscript V) performance). The average ranks refer to the predictive (testing) model performance averaged over the 15 experimental data sets.

training data exhibit significantly better descriptive/training performance than the ones using a separate validation dataset to select ensemble constituents. Together with the results in Figure 7.2 these show a clear case of overfitting — while being superior on the training data, regular selection leads to inferior predictive performance as compared to validation-based selection. This is in line with our previous findings for bagging, boosting and RLS (Simidjievski et al., 2015b, 2016), where the ensembles with constituents chosen on the basis of their training performance clearly overfit — while having significantly better performance on the training data, these ensembles have worse predictive performance on unseen data.

Next, we focus on choosing the optimal number of base models in the ensembles of process-based models. To this end, we compare the predictive performance of ensembles

Table 7.2: Comparison of the descriptive performance (RRMSE on the testing set) of the two methods for choosing ensemble constituents (V=validation, T=training).

Case	Ens.10$_V$	Ens.25$_V$	Ens.50$_V$	Ens.10$_T$	Ens.25$_T$	Ens.50$_T$
B1	0.257	0.262	0.262	**0.177**	0.174	0.177
B2	0.302	0.268	0.279	0.153	**0.150**	0.156
B3	0.374	0.362	0.371	0.260	**0.258**	0.274
B4	0.402	0.359	0.374	0.273	**0.256**	0.263
B5	0.341	0.321	0.314	0.166	**0.150**	0.152
K1	0.948	0.963	0.981	0.390	**0.387**	0.393
K2	0.815	0.788	0.785	**0.673**	0.678	0.683
K3	0.435	0.447	0.427	0.340	0.339	**0.333**
K4	0.671	0.684	0.677	**0.263**	0.271	0.265
K5	0.440	0.460	0.452	0.304	0.301	**0.295**
Z1	0.502	0.562	0.543	0.373	**0.343**	0.345
Z2	0.581	0.594	0.587	0.468	**0.448**	0.454
Z3	0.790	0.806	0.807	0.672	**0.636**	0.645
Z4	0.776	0.777	0.779	0.718	**0.702**	0.710
Z5	0.763	0.759	0.758	0.501	**0.491**	0.493

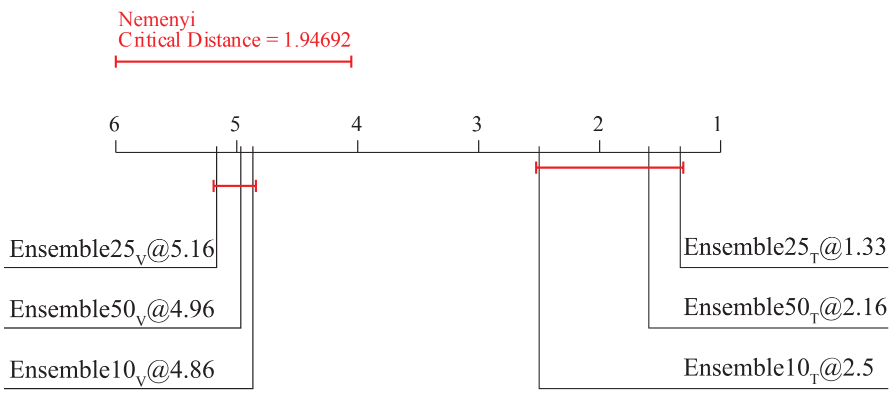


Figure 7.2: Average ranks of ensembles with 10, 25 and 50 constituents combined by averaging and selected differently (based on their train (subscript T) or validation (subscript V) performance). The average ranks refer to the descriptive (training) model performance averaged over the 15 experimental data sets.

consisting of 10, 25 and 50 base models, whose predictions are combined by averaging. The Friedman-Nemenyi diagram, presented in Fig 7.3 together with the results previously presented in Table 7.1, shows that the ensembles containing 25 ensemble constituents lead to the best performance. Note, however, that the observed difference in performance among ensembles with different numbers of base models is not statistically significant.

Finally, we focus on the design decision concerning the most appropriate method for combining the simulations of the base models in the ensemble. We compare the performance of four methods commonly used in learning ensembles of regression models: average, weighted average, median and weighted median. Table 7.3 and Figure 7.4 present the results of the comparison of the actual predictive performance and the respective average ranks of the four methods for combining ensemble simulations with 25 constituents.

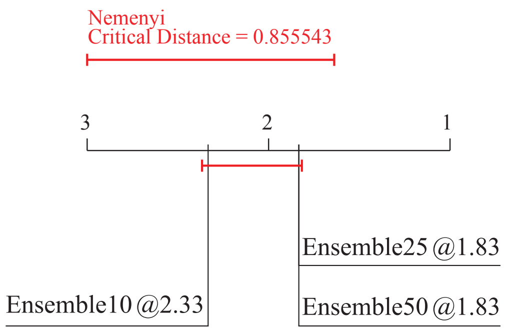


Figure 7.3: Comparison of the average ranks for predictive performance with different ensemble sizes. Average ranks of ensembles that include 10, 25, and 50 base models in terms of predictive performance averaged over the 15 experimental data sets.

In most cases, the simplest method, i.e. *average*, outperforms the other three. However, the differences in performance among all the methods are not significant. Given this, and following the parsimony principle, we can conclude that the most appropriate method for combining the simulations of the ensemble constituents is the simple average.

Table 7.3: Comparison of the predictive performance of the BRLS method with 25 constituents combined by four different methods.

Case	Average	Weighted Average	Median	Weighted Median
B1	1.102	1.095	1.091	**1.089**
B2	**1.237**	1.241	1.262	1.262
B3	0.946	**0.901**	0.995	0.985
B4	0.767	0.746	**0.728**	0.731
B5	0.627	0.621	**0.604**	0.606
K1	**0.639**	**0.639**	0.664	0.667
K2	**0.830**	0.973	1.021	0.905
K3	**0.735**	0.740	0.816	0.817
K4	**0.764**	0.765	0.767	0.768
K5	0.889	0.862	0.835	**0.830**
Z1	1.461	1.422	**0.889**	0.898
Z2	0.844	0.833	**0.800**	0.792
Z3	0.901	**0.898**	0.913	0.922
Z4	**1.005**	**1.005**	1.021	1.021
Z5	**0.999**	1.011	1.004	1.006

In summary, based on the performed experiments, we make the following design decisions related to learning ensembles of process-based models with bagging of random library subsamples. First, the ensemble constituents are chosen based on their performance on a separate validation set, as they exhibit better performance than the ones chosen based on their performance on the original training data set. Second, the ensembles should consist of a relatively small number (25) of base models whose predictions should be combined by using the simple average method. Although the latter conclusion is based on results

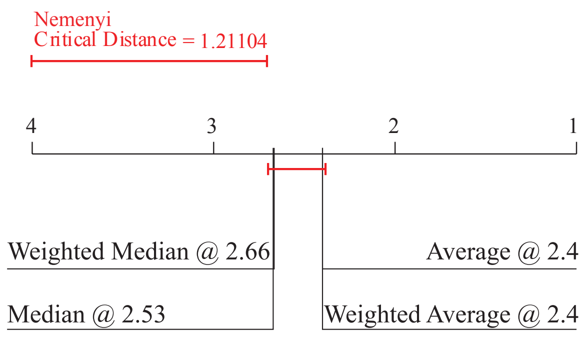


Figure 7.4: Average ranks of the four methods for combining the simulations of base models (average, weighted average, median and weighted median) in terms of predictive performance averaged over the 15 experimental data sets.

which are not statistically significant, we follow the parsimony principle. In the remaining experiments, we use these algorithm settings to learn ensembles with the BRLS method.

7.2 The Performance of Ensembles of Process-Based Models Learned with the Bagging of Random Library Subsamples Method

In our last set of experiments, we assess the predictive performance of the ensembles learned with the bagging of random library subsamples method with the optimal design choices determined above. We compare their performance to the performance of an individual models, and ensembles learned with the RLS method with 10 constituents, bagging and boosting with 25 constituents, all combined with the averaging method.

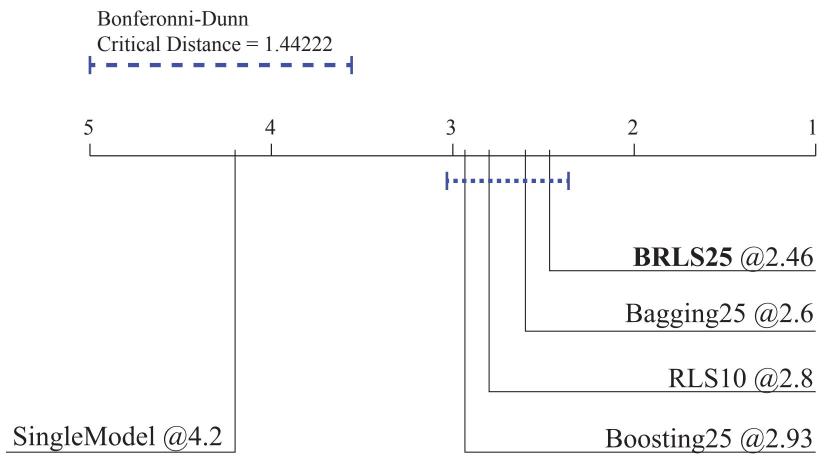


Figure 7.5: Average ranks of ensembles constructed by BRLS to the performance of: a single model and three types of ensembles constructed by bagging, boosting and RLS. The ranks of the models in terms of their predictive performance are averaged over the 15 data sets.

Fig 7.5 depicts the Friedman-Bonferonni-Dunn comparisons of the average ranks across the fifteen data sets. The results of the test show that the BRLS ensembles *significantly* outperform single models. This result once more confirms the central hypothesis of this thesis that ensembles of process-based models improve the predictive performance over single process-based models. Note, however, that their predictive performance is slightly (not significantly) better than the performance of the other three proposed methods for learning ensembles of PBMs.

While Fig 7.5 depicts the comparison of the methods in terms of their average ranks, Table 7.4 reports the actual performances of the single models and the four proposed ensembles of PBMs. Note that the results reported in the table confirm the superiority of BRLS ensembles over a single model: The BRLS method outperforms the single model in 12 cases out of 15. Moreover, in 6 out of these 12, BRLS is the best performing algorithm, overall. However, in the remaining three cases (B2, K5 and Z1) it underperforms. Note that in two of these cases (B2 and Z1), all four algorithms have worse predictive performance than the performance of a single model, whereas in the remaining case (K5) the difference in performance between the single model and BRLS is minor. All things considered, in terms of predictive performance, the BRLS ensembles significantly improve the predictive performance over single models, and are (not significantly) better than the ensembles of process-based models learned with the remaining three methods.

Table 7.4: Comparison of the predictive performance of a single model with that of ensembles of PBMs learned with BRLS, bagging, boosting and RLS.

Case	SingleModel	Bagging	Boosting	RLS	BRLS
B1	5.184 (5)	1.073 (2)	**1.065** (1)	1.103 (4)	1.1 (3)
B2	**0.938** (1)	1.085 (3)	0.947 (2)	1.143 (4)	1.237 (5)
B3	1.042 (5)	0.968 (4)	0.941 (2)	**0.794** (1)	0.946 (3)
B4	2.84 (5)	0.76 (2)	0.791 (4)	**0.75** (1)	0.767 (3)
B5	0.842 (3)	0.859 (5)	0.698 (2)	0.858 (4)	**0.627** (1)
K1	7.73 (5)	0.736 (2)	0.925 (4)	0.756 (3)	**0.639** (1)
K2	328.138 (5)	1.429 (3)	1.642 (4)	0.878 (2)	**0.83** (1)
K3	0.932 (5)	0.867 (3)	0.923 (4)	0.827 (2)	**0.735** (1)
K4	0.777 (4)	0.772 (3)	**0.705**(1)	0.8 (5)	0.764 (2)
K5	0.792 (4)	0.734 (2)	0.781 (3)	**0.732** (1)	0.889 (5)
Z1	**0.744** (1)	0.777 (2)	0.89 (4)	0.797 (3)	1.461 (5)
Z2	1.323 (5)	0.883 (2)	0.893 (3)	0.948 (4)	**0.844** (1)
Z3	29.463 (5)	0.934 (3)	1.001 (4)	**0.881** (1)	0.901 (2)
Z4	27.593 (5)	**0.964** (1)	0.982 (2)	1.011 (4)	1.005 (3)
Z5	1.489 (5)	1.113 (2)	1.403 (4)	1.130 (3)	**0.999** (1)

7.3 Predictive Performance and its Relation to Diversity

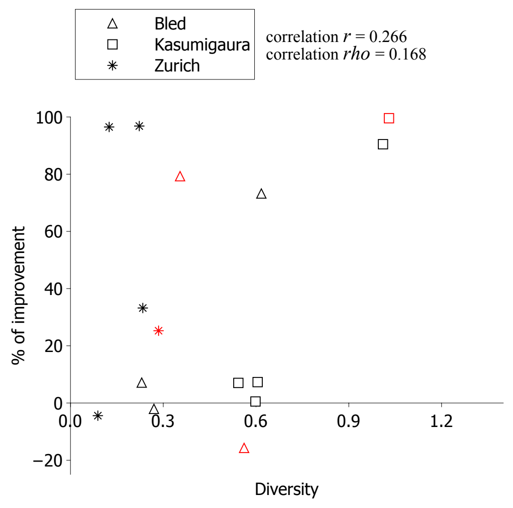
In this section, we present the analyses of the last set of experiments performed for this thesis. Here, we explore the relation of the observed improvement in predictive performance of the ensembles of process-based models to the diversity of the simulations of their constituents. We first measure the relative improvement of the performance obtained by using an ensemble instead of a single process-based model. Then, we measure the diversity of base models in the ensemble. Finally, we try to correlate these two.

Table 7.5 and Figure 7.6 summarize the results of these experiments. First, the results presented in Table 7.5 reconfirm our previous finding that bagging outperforms single models in all but three data sets, B2, B5 and Z1. The gain in performance can be substantial and reach up to 73% for Lake Bled, 90% for Lake Kasumigaura and 96% for Lake Zurich. In the case of boosting, the improvement over a single model is also substantial. The boosting method outperforms single models in the majority of the cases, i.e., in all but two (B2 and Z1). The improvement reaches up to 72% for the case of Lake Bled, up to 88% for Lake Kasumigaura and 96% for Lake Zurich. Similar behavior is observed also for the RLS method. It outperforms the single model in most of the experiments (up to 73% for the case of Lake Bled, up to 99% for Lake Kasumigaura and up to 28% for Lake Zurich). However it also underperforms in 4 cases (B2, B5, K4 and Z1). Finally, the BRLS method, which is also on average the best performing method, outperforms the single model in 12 out of 15 cases (except for B2, K5 and Z1). Here the improvement of predictive performance goes up to 73% for Lake Bled, 99% for Lake Kasumigaura and up to 96% for Lake Zurich. However, in the two cases of B2 and Z1, the single model outperforms every ensemble method proposed. Note that the improvements highlighted above relate to the cases when the ensembles also outperform the average predictor, i.e., have $ReRMSE$ bellow 1.

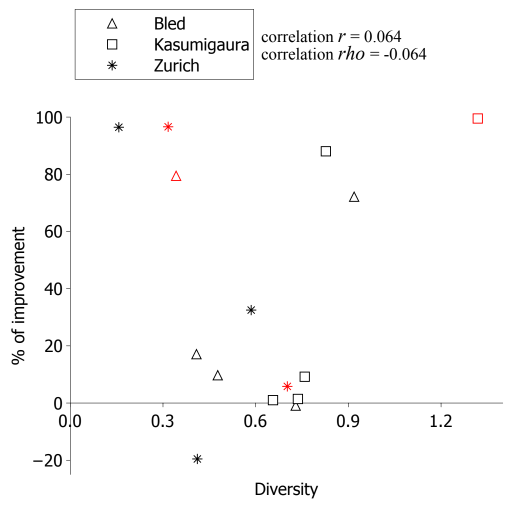
Finally, we observe a varying degree of diversity between ensemble constituents for different data sets—diversity varies from 0.112 to 1.816. The scatter plots in Figure 7.6, for the methods of bagging, RLS and BRLS show weak positive correlation between the ensemble diversity and relative improvement of performance, whereas for the method of boosting there is no correlation. First, the measured Pearson correlation coefficient of 0.266 for bagging, 0.248 for RLS, 0.136 for BRLS and 0.064 for boosting is neither high nor significant. Additionally, neither the measured Spearman's correlation coefficient of 0.168 for bagging, 0.161 for RLS, 0.103 for BRLS and -0.064 for boosting showed significant monotone relation between the ensemble diversity and relative improvement of performance.

Table 7.5: Diversity of the base models and the percentage of relative improvement of the ensemble error over the error of a single model learned for each of the fifteen experiments. In cases highlighted with an asterisk (*) the ensembles' performances are low ($ReRMSE$ above 1).

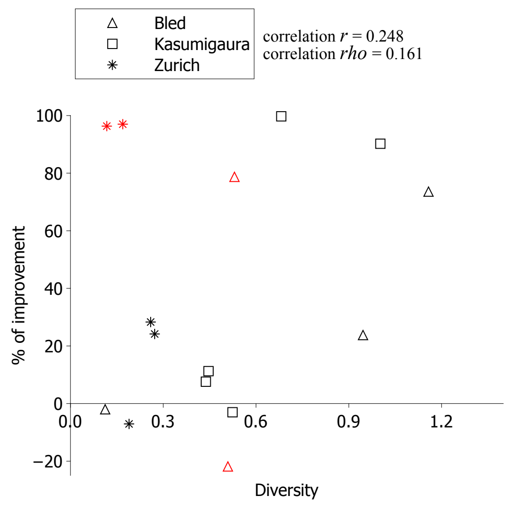
Case	Bagging		Boosting		RLS		BRLS	
	Diversity	Improvement	Diversity	Improvement	Diversity	Improvement	Diversity	Improvement
B1	0.354	79.3*	0.342	79.5*	0.53	78.7*	0.763	78.8*
B2	0.561	-15.7*	0.729	-0.9	0.509	-21.8*	0.235	-31.9*
B3	0.23	7.1	0.477	9.7	0.946	23.8	0.494	9.3
B4	0.617	73.2	0.919	72.2	1.158	73.6	1.000	73.0
B5	0.27	-2.1	0.408	17.1	0.112	-2.0	0.834	25.4
K1	1.01	90.5	0.827	88.0	1.002	90.2	1.259	91.7
K2	1.03	99.6*	1.319	99.5*	0.682	99.7	1.816	99.7
K3	0.543	7.0	0.656	1.0	0.446	11.3	0.967	21.2
K4	0.598	0.5	0.759	9.2	0.524	-3.0	0.379	1.6
K5	0.605	7.3	0.737	1.4	0.438	7.6	0.793	-12.3
Z1	0.089	-4.4	0.411	-19.6	0.189	-7.1	1.096	-96.3*
Z2	0.234	33.2	0.585	32.5	0.259	28.3	0.273	36.2
Z3	0.223	96.8	0.317	96.6*	0.169	97.0	0.241	96.9
Z4	0.125	96.5	0.157	96.4	0.118	96.3*	0.242	96.4*
Z5	0.285	25.3*	0.702	5.8*	0.272	24.2*	0.301	32.9
Pearson r	0.266		0.064		0.249		0.136	
Spearman ρ	0.168		-0.064		0.161		0.103	



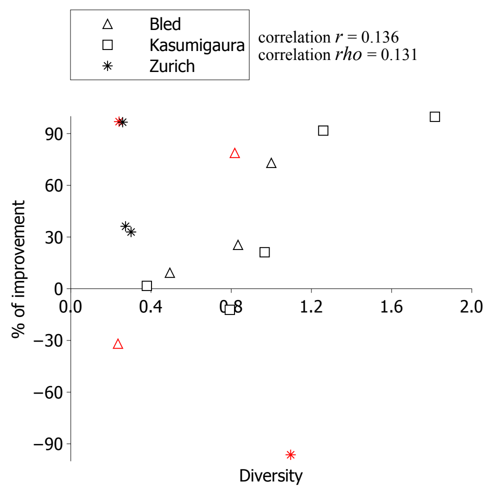
(a) Bagging



(b) Boosting



(c) Random Library Subsamples



(d) Bagging of Random Library Subsamples

Figure 7.6: Scatter plots depicting the correlation between the diversity of the base-model predictions and the relative error improvement between a single model and an ensemble for fifteen data sets. The points colored in red denote the cases where ensembles perform worse than the average predictor ($ReRMSE$ above 1).

Part IV

Conclusions

Chapter 8

Conclusions

In this thesis, we have developed and empirically evaluated a novel methodology for learning ensembles of process-based models. The proposed methodology extends the state-of-the-art paradigm of equation discovery, which is predominately employed in a descriptive setting, towards predictive modeling. The state-of-the-art approaches to automated modeling of dynamic systems, developed within the area of equation discovery, focus primarily on establishing explanatory models of the observed system. In this context, the obtained models are employed for analysis of behaviors encapsulated in the data used for learning them. However, their ability to predict the systems' behavior beyond the time-period captured in the training data is poor. This directly relates to a well known phenomena in machine learning referred to as overfitting: While process-based models are very successful at accurately explaining the observed behavior of the modeled system, they have limited potential for accurate long-term predictions. Considering these shortcomings of the state-of-the art process-based modeling approaches, we propose a novel methodology which follows the basic principles of ensemble learning, and translates them into a methodology for modeling dynamic systems. We illustrate the utility of learning ensembles of process-based models on several tasks of long-term predictive modeling of dynamic systems.

The proposed methodology extends the scope of the traditional ensemble learning paradigm towards modeling dynamic systems. It also extends the state-of-the-art equation discovery methods towards improving their generalization power, subsequently providing accurate simulation of the future behavior of the modeled dynamic systems. To this end, the proposed methodology employs four different methods for constructing ensembles of process-based models.

More specifically, the implementation involves adapting four well-established methods for constructing homogeneous ensembles. Homogeneous ensembles consist of models constructed using the same machine learning algorithm on different samples of the training data and/or knowledge. These methods include learning ensembles of process-based models by (1) bagging, (2) boosting, (3) learning from random library subsamples (RLS) and (4) bagging of random library subsamples (BRLS). Furthermore, we propose a novel method for simulating (combining) such ensembles, robust enough to handle sensitive and divergent trajectories resulting from long-term simulations of process-based models. Finally, to properly evaluate the predictive performance of the proposed methods, and process-based models in general, we have developed an extensive empirical evaluation framework. For the purpose of this dissertation, the experimental analyses within this framework involve a series of predictive tasks of modeling dynamics in real-world aquatic ecosystems.

The results of the performed experimental evaluation provide important evidence that ensembles of process-based models, with the optimal design properties identified, provide much more accurate predictions of concentrations of species in aquatic ecosystems than a single process-based model. These results confirm the first two hypotheses of this dissertation: *Traditional machine learning ensemble methods can be adapted to the task of learning process-based models for modeling dynamic systems. The resulting methods yield improved predictive performance as compared to a single process-based model.*

These results are consistent over experiments with data from three real-world aquatic ecosystems: Lake Bled in Slovenia, Lake Kasumigaura in Japan, and Lake Zurich in Switzerland. While single models struggle with achieving the performance of the baseline predictor (that predicts constant species concentrations at the level of their average), the ensembles of process-based models lead to accurate predictions of population dynamics over long prediction periods, e.g., one season (year) in advance. When compared to previous results obtained in the domain of population dynamics (Atanasova, Todorovski, Džeroski, Remec, et al., 2006; Atanasova, Recknagel, et al., 2006), this is a non-trivial improvement of predictive performance over the state-of-the-art models of population dynamics.

We observed a varying degree of diversity between ensemble constituents for different data sets. Moreover, the measured correlation coefficients did not show a significant linear and/or monotonic relationship between the improvement in predictive performance and the intra-ensemble diversity. Based on these results, we do not have enough evidence to confirm the last stated hypothesis of this thesis, that is, *the diversity between ensemble constituents is highly correlated with the predictive performance of an ensemble.*

In the remainder of this section, we will first discuss the findings of the presented study in both the context of related research and the limitations of the proposed methodology. Next, we will outline the important contributions of this thesis. Finally, we suggest possible directions for further work.

8.1 Discussion

8.1.1 On the performance of ensembles

Based on the results of the empirical evaluation, we identified the optimal design decisions for learning ensembles of process-based models for the task of predicting population concentration in aquatic ecosystems. First, when learning such ensembles, one should use a separate validation data set in addition to the training one when learning the base-models included in the ensemble. This yields more robust ensembles, which in turn substantially improve the predictive performance. For combining the simulation of constituent process-based models, one should use the simplest combining scheme, i.e., averaging.

The optimal ensembles of PBMs consist of a relatively small number of constituent models, ranging from 10 for the RLS method to 25 for both bagging and boosting. However, the process-based models, when simulated in a predictive setting, often produce divergent simulations, i.e., simulations where the systems variables leave their plausible ranges. Therefore, when simulating ensembles of PBMs, one should explicitly handle this kind of behavior of the base models. The proposed method in this thesis uses the provided domain knowledge on system variable ranges to discard the invalid behaviors from the resulting ensemble prediction. This can be viewed as a dynamic (domain-knowledge based) approach to ensemble simulation.

Figure 8.1 presents a summary of the empirical results from the experimental evaluation performed in this thesis in terms of predictive performance of the proposed ensemble methods and a single process-based model. Different methods on the x-axis are sorted with respect to increasing efficiency of the learning algorithms. From this Figure, and the actual predictive performances presented in Table 7.4, we can confirm the superiority

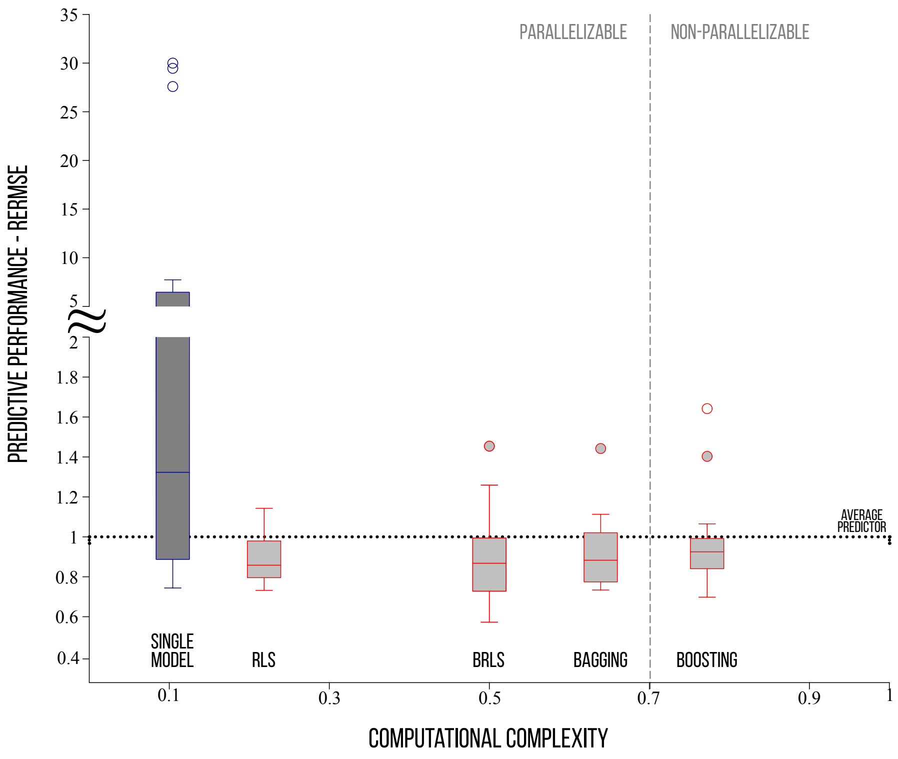


Figure 8.1: An overview of the proposed ensemble methods (red) for learning ensembles of process-based models and a single process-based model (blue) in terms of predictive performance (*ReRMSE* obtained from the experimental evaluation) sorted with respect to increasing efficiency of the learning algorithms. The labels on the x-axis relate to relative computational time.

of ensembles to a single model. The ensemble methods are far more robust than single models, which severely under-perform in several cases.

However, choosing between the proposed ensemble methods is a non-trivial task. Therefore, we need to decompose this task based on three criteria: average predictive performance, robustness and computational efficiency. Based on the first criterion, the ensembles learned with *BRLS* provide the best results. The several statistical tests performed throughout this manuscript, which consider average ranks, indicate that the BRLS method, besides significantly improving the predictive performance compared to a single model, outperforms also the rest of the ensemble methods although not significantly. Notice that even though in the majority of the experiments this method considerably outperforms the single model (in 8 cases this improvement is even greater than 30%), in 3 cases the single model performs better. This method however is closely followed by *bagging*, which also in the majority of cases significantly outperforms the single models.

Next, based on the second criterion, *boosting* is the most robust method. Overall, boosting ensembles exhibit a stable behavior, which translates to predictive improvement in performance over the single model in cases when the other ensemble methods fail. The two cases when the single process-based models performs better, in one case (B2) this difference is less than 1%. For these two cases, both in terms of boosting process-based models and ensembles of process-based models in general, we relate to the studies of Joshi, Agarwal, and Kumar (2002) and Breiman (1996a). In the former, the authors point out that the performance of boosting is correlated with the performance of the base learners. This means that when a single model obtained exhibits very good predictive performance, the boosting weak models can lead only to similar or worse performance. Similarly, this phenomenon is also pointed out in the latter study for the case of bagging. More specifically, if the single model is close to the limits of a attainable accuracy for a specific learning problem, the ensembles rarely yield improved predictive performance.

Computational efficiency, beside predictive performance and robustness, is an important criterion for selecting the most suitable ensemble methods. In this context, when choosing among the proposed ensemble methods based on computational efficiency and predictive performance, the method for learning ensembles from *random library subsamples* (RLS) is the clear winner. The computational efficiency of the RLS method allows for applications to domains where a rich library of domain knowledge is available leading to a large space of candidate models. In such cases, the iterative methods of bagging, boosting and even BRLS are not applicable due to the prohibitively high computational costs. To apply them, one needs to handcraft the library of domain knowledge, omit modeling alternatives, and thus simplify the space of candidate model structures. In contrast, RLS, in addition to being more efficient, often leads to models with better predictive performance as compared to the models obtained with the other ensemble methods.

8.1.2 On the influence of diversity

Machine learning literature provides various frameworks for explaining the performance improvement gained using ensemble methods. Recall from Section 2.2.4, where we stated that while the results on the positive influence of ensemble constituents' diversity on the performance are inconsistent, there is a general consensus that bias-variance-covariance decomposition of the predictive error allows for plausible explanation. To properly investigate this issue in the context of process-based models, we take note of their similarity with regression models, which will stand as a proxy for illustrating the decomposition of the generalization error. For a single model for the task of regression f for an input x, the generalization error of the estimator can be broken down into two components: bias and variance. However, according to Hastie, Tibshirani, and Friedman (2003) there is a trade-off between these components, i.e., attempts on decreasing the bias may result in increased variance and vise versa. The decomposition of the generalization error is then:

$$\begin{aligned} E\{(f-\hat{y})^2\} &= E\{(f-E\{f\})^2\} + (E\{f\}-\hat{y})^2 \\ &\Longrightarrow var(f) + bias(f)^2 = MSE(f)\,, \end{aligned} \tag{8.1}$$

where $\hat{y}$ denotes the expected value of the target variable and $E\{\cdot\}$ is the expectation operator, and $MSE(f)$ is the mean squared error of the estimator. For an ensemble of regressors with M constituents, the average bias and variance can be computed as:

$$\overline{bias} = \frac{1}{M}\sum_i (E\{f_i\}-\hat{y})\,, \tag{8.2}$$

$$\overline{var} = \frac{1}{M}\sum_i E\{(f_i - E\{f_i\})^2\}\,. \tag{8.3}$$

However, the generalization error when ensembles are considered, according to Ueda and Nakano (1996), has a third component, i.e., the covariance between the ensemble constituents. The average covariance of the base models is then computed as:

$$\overline{covar} = \frac{1}{M(M-1)} \sum_i \sum_{j \neq i} E\{(f_i - E\{f_i\})(f_j - E\{f_j\})\} \tag{8.4}$$

From equations 8.2, 8.3 and 8.4 the *bias-variance-covariance* decomposition of the generalization error of the ensemble is:

$$E\{(\frac{1}{M}\sum_i f_i - \hat{y})^2\} = \overline{bias}^2 + \frac{1}{M}\overline{var} + (1 - \frac{1}{M})\overline{covar} \tag{8.5}$$

From equation 8.5 it can be assumed that the component of covariance has a big influence on the generalization error, that is, the diversity between the ensemble constituents is strongly (negatively) correlated to the generalization error and would substantially influence the overall predictive loss. Ensembles tend to reduce the variance component of the predictive error, while not increasing the bias component at the same time (Brown, Wyatt, Harris, & Yao, 2005). This is what happens in the case of learning ensembles of process-based models: by averaging the predictions of several models learned on the same dataset, we reduce the variance component of the predictive error.

Moreover, the analysis of the measured diversity presented in this thesis relates to the component of covariance in the overall ensemble performance. However, the results provide evidence that the correlation between the covariance among the ensemble process-based constituents and the overall performance gain of the ensembles is very weak. This however is in-line with the findings of Kuncheva and Whitaker (2003), which show that there is not always a strong correlation between the intra-ensemble diversity (covariance) and the predictive performance of the ensemble, therefore the question of performance-diversity trade-off is more problem/task specific than general. Additional analyses, presented by Brown et al. (2005), also demonstrate that the error components of bias, variance and covariance are very related, meaning that focusing at decreasing one of these errors will affect the others in the opposite way.

We conjecture that ensembles of process-based models also reduce their bias. This is due to the fact that averaged process-model simulations can lead to predictions that are out of scope of a single process-based model. Therefore, ensembles of process-based models (and models of dynamic systems in general) have a potential to extend the original space of individual models leading to reduction of the bias component of the predictive error. Nonetheless, validating this hypothesis, by completely and properly decomposing the generalization error and investigating the relation of every component to the overall performance of the ensemble is a challenge that we will consider for further work.

8.1.3 On modeling dynamic systems

The work presented in this thesis follows two different lines of research. First, it extends the state-of-the-art in the paradigm of equation discovery. More specifically, we build upon previous methods for learning process-based models that have proven successful for automated modeling of population dynamics in a number of aquatic ecosystems (Todorovski & Džeroski, 2007; Borrett, Bridewell, Langley, & Arrigo, 2007; Čerepnalkoski et al., 2012; Bridewell et al., 2008). Second, it follows the basic principles of ensemble learning, and translates them into a methodology for modeling dynamic systems.

This work is closely related to that of Bridewell et al. (2005), where the authors use ensemble methods to establish better descriptive models by tackling the over-fitting problem. Their approach is based on integrating the model structures of ensemble constituents into a single model. This model still provides a process-based explanation of the observed system structure, while being more robust in terms of over-fitting observed data. The evaluation of overfitting is performed by a variant of the general cross-validation method, where samples of data are kept out of the training set and are used to estimate the model error. While this method provides estimates of model error on unseen data, these estimates are not related to the predictive performance of the model, i.e., its ability to predict future system behavior beyond the time-period captured in training data.

The studies of Whigham and Recknagel (2001), Cao, Recknagel, Cetin, and Zhang (2008), Gilboa, Friedler, and Gal (2009) are also related to our work, as they employ automated approaches to modeling dynamics of lake ecosystems in a process-based formalism. However, these studies start from a modeling assumption that includes a fixed structure of model equations and employ genetic algorithms to calibrate the values of the fixed model parameters and obtain better simulations. The study of Aleksovski et al. (2015) also relates to our work, since it tackles the tasks of predictive modeling of (discrete) non-linear dynamic systems with machine learning approaches. Here, the modeling problem is first transformed into a non-linear regression approximation problem subsequently addressed by learning fuzzy linear model trees and ensembles of fuzzy linear model trees. The results show that the ensembles improve the performance over the single fuzzy linear trees. While this study focuses on short-term (one-step ahead) prediction of discrete-time dynamic systems, where the value of the time series in the next time-point is predicted; the tasks of long-term prediction of continuous non-linear dynamic systems are not considered nor evaluated.

In a broader sense, this work also relates to a long tradition of accurately predicting behaviors of dynamic systems by combining diverse predictions obtained by perturbing parameters of an individual model. A recent paradigm (Van den Berge et al., 2011; Mirchev et al., 2012), referred to as supermodeling, aims at constructing an ensemble of "imperfect" models, comprised from a set of interconnected (coupled) models, which are integrated simultaneously and are able to exchange information among themselves on a time-step basis. However, the focus of this paradigm is predominantly estimation of the coupling coefficients, whereas the structures of these "imperfect" models, often very complex, are provided a priori by a domain expert.

While the comparative empirical study, presented in this study, is limited to the domain of modeling population dynamics, the proposed methodology to learning ensembles of process-based models is general enough to be applied to any other domain and to any other type of models of dynamic systems. To proceed with process-based modeling in other domains, one has to encode a library of domain-specific modeling knowledge. These are available for various domains including modeling active potentials in neurons (Simidjievski, Todorovski, & Džeroski, 2014), epidemiology and modeling gene regulatory networks (Tanevski et al., 2016), net carbon production (Todorovski, 2003) and protein interactions (Taškova et al., 2011; Tanevski et al., 2015). Some of these applications of process-based modeling show the applicability of the proposed approach to the very active fields in life sciences such as systems biology, systems neuroscience and bioinformatics that deal with numerous identification problems. The predictive ability of the process-based ensembles bears a promise for further applications, since combining predictive models into ensembles have been proved to work well for predictive tasks in various domain.

8.1.4 On ensemble interpretability

In this thesis, for designing the methods for learning ensembles of process-based models, we relate to traditional and well-established methods in machine learning, i.e., bagging (Breiman, 1996a), Adaboost (Freund, 1999), the random subspaces method (Ho, 1998) and bagging of random subspaces (Panov & Džeroski, 2007). One thing that all of these methods have in common is that they lack the ability to produce an interpretable ensemble.

The ensemble methods proposed in this thesis aim at improving the generalization power of process-based models, in terms of achieving predictive performance gain over the state-of-the-art process-based modeling approaches. However, when learning ensembles of process-based models, they also inherit the same limitation: a trade-off between two conflicting requirements – predictive performance and interpretability. The increase of predictive accuracy comes at the cost of losing the interpretability of the learned ensemble model. Nevertheless, interpretability is an inherit deficiency of ensembles in general, given that they aggregate a set of models in order to be constructed.

Several attempts are made that tackle this challenge. In the context of learning ensemble of process-based models, probably the most appropriate approach would be learning a meta-model. Here, the idea is constructing a new process-based model out of the ensemble constituents, either by combining components of their structures or learning additional parameters which connects them. The former closely relates to the study of (Bridewell et al., 2005), where the authors propose a method which integrates the model structures of the ensemble constituents into a single complex meta-model. The latter refers to transforming the task of a posteriori combination of the trajectories of the individual ensemble constituents into contracting a system of Ordinary Differential Equations which would be simulated simultaneously. In this context, the approach of supermodeling would most appropriately fit in. However, further investigations are required to determine which is the most appropriate approach to learn interpretable ensembles of process-based models, and more importantly, without harming their predictive performance.

8.2 Contributions to Science

This thesis contributes to several important aspects of the process-based modeling paradigm. The majority of these findings are published in several journal and conference publications, some of them are included in this manuscript. The complete list of publications is given in the appendix titled *Bibliography*. In the following subsections, we summarize the main scientific contributions of the work presented in this thesis.

8.2.1 A novel methodology for learning ensembles of process-based models

In this thesis we address the task of learning ensembles of process-based models by designing, implementing and evaluating the appropriate methodology. The developed methodology is general and modular, which allows for various extensions for adapting different ensemble methods to the particular context of learning process-based models. This methodology is the main contribution of this thesis, since it extends the scope of current process-based modeling approaches to the task of learning ensembles of process-based models. To this end, the proposed ensemble methodology consists of four different algorithms for learning homogeneous ensembles of process-based models, as well as an algorithm for simulating them.

This contribution is also important in the wider context of the ensemble learning paradigm, applied in the context of time-series predictive tasks. While such ensembles have a narrow focus on short-term prediction tasks, where the value of the time series at the next time point is predicted, ensembles of process-based models provide accurate long-term predictions over many future time points.

8.2.2 An evaluation framework for empirical analysis of the predictive performance of process-based modeling methods

We designed and performed an empirical evaluation framework for estimating the predictive performance of process-based modeling methods. In terms of the proposed ensemble methodology, this framework allows for identification of the optimal design properties of the different ensemble methods proposed in this thesis. Moreover, it provides a thorough and fair evaluation of the predictive performance of the proposed methods. In this thesis, the experimental evaluation involves a series of predictive tasks of modeling population dynamics in real-world aquatic ecosystems.

Note, however, while the evaluation framework in the thesis has only been used in the limited context in the domain of ecological modeling, it is easily applicable towards other application domains, both real-world and artificial. Therefore, this contribution is also important in the wider context of the process-based modeling paradigm. While the state-of-the-art approaches focus mainly on measuring the descriptive performance of the obtained process-based models or perform a cross-validation, here we propose a standardized procedure for measuring performance of models when applied to tasks of modeling future behaviors beyond the time-period captured in training data.

8.2.3 Improved long-term prediction as compared to state-of-the-art approaches to process-based modeling of dynamic systems

In this context, we compared the predictive performance of different ensembles of process-based models to the performance of a single process-based model. We conducted a thorough empirical evaluation on the task of modeling population dynamics in aquatic ecosystems. The considered case studies addressed the task of modeling phytoplankton growth, a complex non-linear dynamic process, in three different real-life aquatic ecosystem domains. Based on the performed empirical evaluation, we also identified the main design decisions that need to be made when learning ensembles of process-based models. In general, all proposed ensemble methods, following the respective designs, yield to significantly better predictive performance compared to a single process-based models.

This contribution is also important in the wider context of ensembles for time-series forecasting. While forecasting ensembles have a narrow focus on short-term prediction tasks, where the value of the time series at the next time point is predicted, ensembles of process-based models provide accurate long-term predictions over many future time points.

8.2.4 Accurate long-term predictions of population dynamics in real-world aquatic ecosystems

Finally, the work presented in this thesis provides an important contribution to the domain of ecological modeling. The results of the performed experimental evaluation confirm that ensembles of process-based models provide (1) logically plausible and (2) accurate predictions of concentrations of species in aquatic ecosystems. Moreover they tend to improve the predictive performance compared to the one of a single process-based model. This is also a significant improvement of predictive performance over the state-of-the-art models of population dynamics, which is limited at providing an accurate prediction of the future behavior of the observed system (Atanasova, Todorovski, Džeroski, Remec, et al., 2006; Atanasova, Recknagel, et al., 2006).

8.3 Further Work

We have identified several limitations of our methodology, which can be addressed by following several directions for further work. First, note that the experiments performed in this thesis are limited to modeling population dynamics in three lake ecosystems. An immediate continuation of the work presented here is to investigate the generality of the proposed methodology and extend the scope of learning ensembles of process-based models of population dynamics to other aquatic environments, such as marine and river ecosystem. Next, we plan to investigate whether the findings of this thesis are also consistent when applied in an artificial setting, for predictive tasks of synthetic dynamic systems. Here, the artificial data is first generated by simulating known process-based models previously learned for a task of modeling population dynamics. The goal is then predictive reconstructing of the behavior of the known model under various levels of noise, therefore evaluating the robustness of the proposed ensemble methodology. Finally, we also intend to analyze the applicability of proposed methodology to different scientific domains, such as systems biology and systems neuroscience. Here, however, an additional challenge lies in developing a suitable and extensive library of domain knowledge and obtaining valid measurements of the systems.

The second direction considers the ensemble performance-diversity enigma. The results from the experimental evaluation showed that there is no significant linear relation between the diversity of the ensemble constituents and the overall performance of the ensemble. Nevertheless, we intend to explore this problem further. First, we plan to follow ideas from Geman et al. (1992), Brown et al. (2005) and investigate this relation in terms of *bias-variance-covariance* decomposition of the predictive error of the ensemble. In order to do this properly, this will include additional (100) tasks of learning ensembles of process-based models, so any randomness in the learning procedure will be alleviated. Second, in this thesis we measure the diversity between the ensemble constituents at the trajectory level. However, process-based models besides this quantitative property also encompass qualitative structural property. In this context, we plan to investigate whether the structural diversity between the base models influences the overall predictive performance of the ensemble.

In this thesis we proposed four methods for learning ensembles of process-based models. The results of the evaluation of predictive performance showed that boosting process-based models, can yield to overall stable and accurate predictions. However, in terms of computational efficiency, learning ensembles of process-based models with boosting, especially when the library of domain knowledge leads to a large space of candidate models, is prohibitive. On the other hand, the most efficient ensemble method, i.e. the RLS method, greatly benefits from learning each ensemble constituents on smaller subset of the background knowledge. One direction for future work would be utilizing the properties of these two algorithms, i.e., robustness and computational efficiency, by combining boosting and RSM, therefore learning ensembles with boosting random library subsamples. We conjecture that the new algorithm will still have satisfiable predictive accuracy and robustness, while being learned in a reasonable computational time-frame.

In general the ensemble methods proposed in this thesis add substantial computational overhead compared to learning a single process-based model. This overhead primarily relates to the space of model structures instantiated from the library of domain knowledge. To this end, the process od enumerating the model structures implemented in ProBMoT is exhaustive, which means that in practice this results in a very large space of pleasurable models considered in each ensemble iteration. To tackle this limitation, we plan to implement a heuristic search strategy over the space of candidate structures, which in turn will result in obtaining the most suitable models structures for the problem at hand within a reasonable time frame. This allows us to tackle more complex modeling tasks, where the iterative ensemble methods, such as bagging and boosting, will benefit the most.

Finally, considering the interpretability of the ensembles of process-based models, we plan to follow ideas from (Bridewell et al., 2005) and improve our methodology by incorporating understandable structure into the resulting ensemble. In a similar context, we intend to extend our methodology towards learning interactive ensembles of models of dynamic systems (Mirchev et al., 2012). In contrast to the proposed ensembles methods, where the base models are learned and simulated independently and combined afterwards, within the interactive ensembles, the base models are allowed to share and interchange information both during the learning and the simulation phase. In this context, one can learn the constituent models, their interconnections or both.

Appendix A

Library of Domain Knowledge for Modeling Population Dynamics in Aquatic Ecosystems

For the tasks of modeling population dynamics in the three real-world lake ecosystems presented in this thesis, we used the library of domain knowledge developed by Atanasova, Todorovski, Džeroski, and Kompare (2006), which is presented in the remainder of this chapter.

```
library AquaticEcosystem;

// ENTITIES

template entity EcosystemEntity {
    vars:
    conc {aggregation:sum; unit:"kg/m3̂"; range:<0,20>};
}
template entity Population:EcosystemEntity {
    vars:
        tempGrowthLim{aggregation:product},
        tempRespLim{aggregation:product},
        tempMortLim{aggregation:product},
        tempExcLim{aggregation:product},
        tempSedLim{aggregation:product};
}
template entity PrimaryProducer:Population{
    vars:
        nutrientLim{aggregation:product},
        lightLim{aggregation:product},
        growthRate;
    consts:
        maxGrowthRate {range:<0.05,3>; unit:"1/(day)"};
}
template entity Zooplankton:Population{
    vars:
        phytoLim{aggregation:sum},
        phytoSum{aggregation:sum};
    consts:
        maxFiltrationRate {range:<0.01,15>; unit:"m3/(mgZoo*day)"},
        assimilationCoeff {range:<0,20>; unit:"mgZoo/(mgAlgae)"};
}
template entity Nutrient:EcosystemEntity{
    consts:
        halfSaturation {range:<0,15>; unit:"mg/l"},
        alpha {range:<0,20>; unit:"mgAlgaeBiomass/mgZooBiomass"};
}
template entity Environment {
    vars:temperature,light,flow;
    consts:volume,depth,area;
}

// PROCESSES

template process NutrientPrimaryProducerInteraction(pp:PrimaryProducer,ns:Nutrient<1,20>,
    env:Environment ){
    processes:
    LightInfluence(pp,env), NutrientInfluence(pp,<n:ns>),
    Growth(pp,ns,env),RespirationPP(pp,ns,env);
}
// Temperature Growth Influence
template process TempGrowthInfluence(pop:Population,env:Environment){}
template process TempGrowthLim:TempGrowthInfluence {
    consts:
        refTemp {range:<10,22>},
        minTemp {range:<0,6>},
        optTemp {range:<15,25>};
}
template process Linear1TempGrowthLim:TempGrowthLim{
    equations:  pop.tempGrowthLim = env.temperature/refTemp;
}

template process Linear2TempGrowthLim:TempGrowthLim{
    equations:
        pop.tempGrowthLim = (env.temperature - minTemp)/(refTemp - minTemp);
}
template process ExponentialTempGrowthLim:TempGrowthLim{
    consts:  theta {range:<1.06,1.13>};
    equations:
        pop.tempGrowthLim = pow(theta, env.temperature - refTemp);
}
// Temperature Respiration Influence
template process TempRespInfluence(pop:Population,env:Environment){}
template process TempRespLim:TempRespInfluence{
    consts:
        refTemp {range:<10,22>},
        minTemp {range:<0,6>},
        optTemp {range:<15,25>};
```

```
}
template processLinear1TempRespLim:TempRespLim{
    equations:  pop.tempRespLim = env.temperature/refTemp;
}
template process Linear2TempRespLim:TempRespLim{
    equations:
        pop.tempRespLim = (env.temperature − minTemp)/(refTemp − minTemp);
}
template process ExponentialTempRespLim:TempRespLim{
    consts:  theta {range:<1.06,1.13>};
    equations:
        pop.tempRespLim = pow(theta, env.temperature − refTemp);
}
// Temperature Mortality Influence
template process TempMortInfluence(pop:Population,env:Environment){}
template process NoTempMortLim:TempMortInfluence{
    equations:  pop.tempRespLim = 1;
}
template process TempMortLim:TempMortInfluence{
    consts:
        refTemp {range:<10,22>},
        minTemp {range:<0,6>},
        optTemp {range:<15,25>};
}
template process Linear1TempMortLim:TempMortLim{
    equations:  pop.tempRespLim = env.temperature/refTemp;
}
template process Linear2TempMortLim:TempMortLim{
    equations:
        pop.tempRespLim = (env.temperature − minTemp)/(refTemp − minTemp);
}
template process ExponentialTempMortLim:TempMortLim{
    consts:  theta {range:<1.06,1.13>};
    equations:
        pop.tempRespLim = pow(theta, env.temperature − refTemp);
}
// Temperature Excretion Influence
template process TempExcInfluence(pop:Population,env:Environment){}
template process NoExcMortLim:TempExcInfluence{
    equations:  pop.tempExcLim = 1;
}
template process TempExcLim:TempExcInfluence{
    consts:
        refTemp {range:<10,22>},
        minTemp {range:<0,6>},
        optTemp {range:<15,25>};
}
template process Linear1TempExcLim:TempExcLim{
    equations:  pop.tempExcLim = env.temperature/refTemp;
}
```

```
template process Linear2TempExcLim:TempExcLim{
    equations:
        pop.tempExcLim = (env.temperature − minTemp)/(refTemp − minTemp);
}

template process ExponentialTempExcLim:TempExcLim{
    consts:theta {range:<1.06,1.13>};
    equations:
        pop.tempExcLim = pow(theta, env.temperature − refTemp);
}
// Temperature Sedimentation Influence
template process TempSedInfluence(pop:Population,env:Environment){}
template process NoTempSedLim:TempSedInfluence{
    equations:  pop.tempSedLim = 1;
}
template process TempSedLim:TempSedInfluence{
    consts:
        refTemp {range:<10,22>},
        minTemp {range:<0,6>},
        optTemp {range:<15,25>};
}
template process Linear1TempSedLim:TempSedLim{
    equations:  pop.tempSedLim = env.temperature/refTemp;
}
template process Linear2TempSedLim:TempSedLim{
    equations:
        pop.tempSedLim = (env.temperature − minTemp)/(refTemp − minTemp);
}
template process ExponentialTempSedLim:TempSedLim{
    consts:  theta {range:<1.06,1.13>};
    equations:
        pop.tempSedLim = pow(theta, env.temperature − refTemp);
}
// Light Influence
template process LightInfluence(pp:PrimaryProducer,env:Environment){}
template process LightLim:LightInfluence{}
template process MonodLightLim:LightLim{
    consts:  halfSat {range:<0,200>};
    equations:  pp.lightLim = env.light/(env.light + halfSat);
}
template process OptimalLightLim:LightLim{
    consts:  optLight {range:<100,200>};
    equations:
        pp.lightLim = env.light ∗ exp(−env.light/optLight + 1)/optLight;
}
// Nutrient Influence
template process NutrientInfluence(pp:PrimaryProducer,n:Nutrient){}
template process NutrientLim:NutrientInfluence{}
template process MonodNutrientLim:NutrientLim{
    equations:  pp.nutrientLim = n.conc/(n.conc + n.halfSaturation);
}
template process Monod2NutrientLim:NutrientLim{
    equations:
        pp.nutrientLim = pow(n.conc, 2)/(pow(n.conc, 2) + n.halfSaturation);  }
template process ExponentialNutrientLim:NutrientLim{
    consts:  saturationRate {range:<0,15>};
    equations:
        pp.nutrientLim = 1 − exp(−saturationRate ∗ n.conc);
}
// Growth
template process Growth(pp:PrimaryProducer,ns:Nutrient<1,20>,env:Environment){
    processes:
    TempGrowthInfluence(pp,env),GrowthRate(pp,ns,env);
        equations:
            td(pp.conc) = pp.growthRate ∗ pp.conc,
            td(< n : ns > .conc) = −n.alpha ∗ pp.growthRate ∗ pp.conc;
}
template process GrowthRate(pp:PrimaryProducer,ns:Nutrient<1,20>,env:Environment){}
template process LimitedGrowthRate:GrowthRate{
    equations:
        pp.growthRate = pp.maxGrowthRate ∗ pp.tempGrowthLim
                        ∗pp.lightLim ∗ pp.nutrientLim;
}
// Respiration PP
template process RespirationPP(pp:PrimaryProducer,ns:Nutrient<1,20>,env:Environment){}
template process ExponentialRespirationPP:RespirationPP{
    consts:  respRate {range:<0.0001,2>};
    equations:
        td(pp.conc) = −respRate ∗ pp.conc,
        td(< n : ns > .conc) = respRate ∗ pp.conc;
}
template process TempRespirationPP:RespirationPP{
    processes:  TempRespInfluence(pp,env);
}
template process Temp1RespirationPP:TempRespirationPP{
    consts:  respRate {range:<0.0001,1>};
    equations:
        td(pp.conc) = −respRate ∗ pp.conc ∗ pp.tempRespLim,
        td(< n : ns > .conc) = respRate ∗ pp.conc ∗ pp.tempRespLim;
}
template process Temp2RespirationPP:TempRespirationPP{
    consts:  respRate {range:<0.0001,1>};
    equations:
        td(pp.conc) = −respRate ∗ pow(pp.conc, 2) ∗ pp.tempRespLim,
        td(< n : ns > .conc) = respRate ∗ pow(pp.conc, 2) ∗ pp.tempRespLim;
}
// Mortality PP
template process MortalityPP(pp:PrimaryProducer,env:Environment){
    processes:  TempMortInfluence(pp,env);
}
```

```
template process TempMortalityPP:MortalityPP{
    consts: mortRate {range:<0.0001,2>};
    equations: $td(pp.conc) = -mortRate * pp.conc * pp.tempMortLim$; }
template process Temp2MortalityPP:MortalityPP{
    consts: mortRate {range:<0.0001,2>};
    equations:
        $td(pp.conc) = -mortRate * pow(pp.conc, 2) * pp.tempMortLim$;
}
// Feeds On
template process FeedsOn(zoo:Zooplankton,pps:PrimaryProducer<1,20>,env:Environment){
    processes: TempGrowthInfluence(zoo,env),PhytoLim(zoo,pps);
}
template process FeedsOnFiltration:FeedsOn{
    equations:
        $td(zoo.conc) = zoo.assimilationCoeff * zoo.maxFiltrationRate$
                    $*zoo.tempGrowthLim * zoo.conc * zoo.phytoSum * zoo.phytoLim$,
        $td(< pp : pps > .conc) = -zoo.maxFiltrationRate * zoo.tempGrowthLim$
                    $*zoo.conc * pp.conc * zoo.phytoLim$;
}
template process PhytoLim(zoo:Zooplankton,pps:PrimaryProducer<1,20>){}
template process MonodPhytoLim:PhytoLim{
    consts: halfSaturation {range:<0,20> };
    processes: Summation(zoo,pps);
    equations:
        zoo.phytoLim = $zoo.phytoSum/(halfSaturation + zoo.phytoSum)$;
}
template process Monod2PhytoLim:PhytoLim{
    consts: halfSaturation {range:<0,20> };
    processes: Summation(zoo,pps);
    equations:
        $zoo.phytoLim = pow(zoo.phytoSum, 2)/(pow(zoo.phytoSum, 2) + halfSaturation)$;
}
template process Summation(zoo:Zooplankton,pps:PrimaryProducer<1,20>){
    equations: $zoo.phytoSum =< pp : pps > .conc$;
}
template process RespirationZoo(zoo:Zooplankton,ns:Nutrient<1,20>,env:Environment){
    processes: TempRespInfluence(zoo,env);
}
template process TempRespirationZoo:RespirationZoo{
    consts: respirationRate {range:<0.001,1.5>};
    equations:
        $td(zoo.conc) = -respirationRate * zoo.tempRespLim * zoo.conc$,
        $td(< n : ns > .conc) = respirationRate * zoo.conc$;
}
template process Temp2RespirationZoo:RespirationZoo{
    consts: respirationRate {range:<0.001,1.5>};
    equations:
        $td(zoo.conc) = -respirationRate * zoo.tempRespLim * zoo.conc * zoo.conc$,
        $td(< n : ns > .conc) = respirationRate * pow(zoo.conc, 2)$;
}
```

```
// Mortality Zoo
template process MortalityZoo(zoo:Zooplankton){}
template process ExponentialMortalityZoo:MortalityZoo{
    consts:  mortRate {range:<0.0001,2>};
    equations:  $td(zoo.conc) = -mortRate * zoo.conc;$
}
template process TempMortalityZoo(env:Environment):MortalityZoo{
    consts:  mortRate {range:<0.0001,2>};
    equations:
        $td(zoo.conc) = -mortRate * zoo.conc * zoo.tempMortLim;$
}
template process Temp2MortalityZoo:MortalityZoo{
    consts:  mortRate {range:<0.0001,2>};
    equations:
        $td(zoo.conc) = -mortRate * pow(zoo.conc, 2) * zoo.tempMortLim;$
}

template process HyperbolicMortalityZoo:MortalityZoo{
    consts:  decay {range:<0.001,1.5>};
    equations:
        $td(zoo.conc) = -pow(zoo.conc, 2) * decay/(decay + zoo.conc);$
}
template process SigmoidMortalityZoo:MortalityZoo{
    consts:  decay {range:<0.001,1.5>};
    equations:
        $td(zoo.conc) = -pow(zoo.conc, 3) * decay/(pow(decay, 2) + pow(zoo.conc, 2));$
}
template process ExcretionZoo(zoo:Zooplankton,env:Environment){}
template process TempExceretionZoo:ExcretionZoo{
    consts:  exRate {range:<0.0001,2>};
    equations:
        $td(zoo.conc) = -exRate * zoo.conc * zoo.tempExcLim;$
}
template process Temp2ExceretionZoo:ExcretionZoo{
    consts:exRate {range:<0.0001,2>};
    equations:
        $td(zoo.conc) = -exRate * pow(zoo.conc, 2) * zoo.tempExcLim;$
}
template process Sedimentation(pop:Population,env:Environment){
    processes:  TempSedInfluence(pop,env);
    consts:  sedimentationRate {range:<0.0001,0.5>; unit:"1/(day)"};
    equations:
        $td(pop.conc) = -(sedimentationRate/env.depth) * pop.conc * pop.tempSedLim;$
}
```

References

Agrawal, R., Mannila, H., Srikant, R., Toivonenu, H., & Verkamo, I. A. (1996). Advances in knowledge discovery and data mining. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy (Eds.), (Chap. Fast Discovery of Association Rules, pp. 307–328). Menlo Park, CA, USA: American Association for Artificial Intelligence.

Aleksovski, D., Kocijan, J., & Džeroski, S. (2015). Ensembles of fuzzy linear model trees for the identification of multi-output systems. *IEEE Transactions on Fuzzy Systems.*

Ali, K. M. (1996). *Learning probabilistic relational concept descriptions* (Doctoral dissertation, University of California, Irvine, Irvine, CA, US).

Ali, K. M. & Pazzani, M. (1996). Error reduction through learning multiple descriptions. *Machine Learning*, *24*(3), 173–202.

Allwein, E. L., Schapire, R. E., & Singer, Y. (2000). Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, *1*, 113–141.

Atanasova, N., Recknagel, F., Todorovski, L., Džeroski, S., & Kompare, B. (2006). Computational assemblage of ordinary differential equations for chlorophyll-a using a lake process equation library and measured data of Lake Kasumigaura. In F. Recknagel (Ed.), *Ecological informatics* (pp. 409–427). Springer.

Atanasova, N., Todorovski, L., Džeroski, S., & Kompare, B. (2006). Constructing a library of domain knowledge for automated modelling of aquatic ecosystems. *Ecological Modelling*, *194*(1-3), 14–36.

Atanasova, N., Todorovski, L., Džeroski, S., Remec, R., Recknagel, F., & Kompare, B. (2006). Automated modelling of a food web in Lake Bled using measured data and a library of domain knowledge. *Ecological Modelling*, *194*(1-3), 37–48.

Bakır, G. H., Hofmann, T., Schölkopf, B., Smola, A. J., Taskar, B., & Vishwanathan, S. V. N. (2007). *Predicting structured data.* Neural Information Processing. The MIT Press.

Banfield, R. E., Hall, L. O., Bowyer, K. W., & Kegelmeyer, W. P. (2007). A comparison of decision tree ensemble creation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(1), 173–180.

Barzel, B., Liu, Y.-Y., & Barabási, A.-L. (2015). Constructing minimal models for complex system dynamics. *Nature communications*, *6*(7186).

Bauer, E. & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: bagging, boosting, and variants. *Machine Learning*, *36*(1-2), 105–139.

Baxt, W. G. (1992). Improving the accuracy of an artificial neural network using multiple differently trained networks. *Neural Computation*, *4*(5), 772–780.

Bazell, D. & Aha, D. W. (2001). Ensembles of classifiers for morphological galaxy classification. *The Astrophysical Journal*, *548*(1), 219.

Bell, R. M., Koren, Y., & Chris, V. (2008). *The bellkor 2008 solution to the netflix prize.* Statistics Research Department at AT&T Labs. Florham Park, NJ, US.

Bishop, C. M. (1995). *Neural networks for pattern recognition.* Neural Networks for Pattern Recognition. Clarendon Press.

Bishop, C. M. (2007). *Pattern recognition and machine learning (information science and statistics).* Springer.

Blockeel, H., Raedt, L. D., & Ramon, J. (1998). Top-down induction of clustering trees. In *Proceedings of the 15th International Conference on Machine Learning, ICML '98* (pp. 55–63). Morgan Kaufmann.

Borrett, S. R., Bridewell, W., Langley, P. W., & Arrigo, K. R. (2007). A method for representing and developing process models. *Ecological Complexity*, *4*(1–2), 1–12.

Box, G., Jenkins, G., & Reinsel, G. (1994). *Time series analysis: forecasting and control.* Forecasting and Control Series. Prentice Hall.

Bradley, E., Easley, M., & Stolle, R. (2001). Reasoning about nonlinear system identification. *Artificial Intelligence*, *133*(1–2), 139–188.

Bratko, I. (2000). *Prolog programming for artificial intelligence* (3rd). Addison Wesley.

Brazdil, P. B. & Torgo, L. (1990). Knowledge acquisition via knowledge integration. In B. Wielenga (Ed.), *Current trends in knowledge acquisition.* IOS Press.

Bredeweg, B., Linnebank, F., Bouwer, A., & Liem, J. (2009). Garp3 — workbench for qualitative modelling and simulation. *Ecological Informatics*, *4*(5–6), 263–281.

Breiman, L. (1996). *Bias, variance, and arcing classifiers* (tech. rep. No. 460). Statistics Department, University of California, Berkeley. Berkeley, CA, USA.

Breiman, L. (1996a). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Breiman, L., Friedman, J. H., Stone, C. J., & Olshen, R. A. (1984). *Classification and Regression Trees.* Chapman & Hall, London, UK.

Bridewell, W., Asadi, N. B., Langley, P. W., & Todorovski, L. (2005). Reducing overfitting in process model induction. In *Proceedings of the 22nd international conference on Machine learning, ICML '05* (pp. 81–88). New York, NY, USA: ACM.

Bridewell, W. & Langley, P. W. (2010). Two kinds of knowledge in scientific discovery. *Topics in Cognitive Science*, *2*(1), 36–52.

Bridewell, W., Langley, P. W., Todorovski, L., & Džeroski, S. (2008). Inductive process modeling. *Machine Learning*, *71*, 1–32.

Brown, G., Wyatt, J., Harris, R., & Yao, X. (2005). Diversity creation methods: a survey and categorisation. *Journal of Information Fusion*, *6*(1), 5–20.

Buntane, W. L. (1990). *A theory of learning classification rules* (Doctoral dissertation, School of Computing Science, University of Technology, Sydney, Australia).

Cao, H., Recknagel, F., Cetin, L., & Zhang, B. (2008). Process-based simulation library SALMO-OO for lake ecosystems. part 2: multi-objective parameter optimization by evolutionary algorithms. *Ecological Informatics*, *3*(2), 181–190.

Caruana, R., Niculescu-Mizil, A., Crew, G., & Ksikes, A. (2004). Ensemble selection from libraries of models. In *Proceedings of the 21st International Conference on Machine Learning, ICML '04* (pp. 18–). New York, NY, USA: ACM.

Čerepnalkoski, D. (2013). *Process-based models of dynamical systems: Representation and induction* (Doctoral dissertation, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia).

Čerepnalkoski, D., Taškova, K., Todorovski, L., Atanasova, N., & Džeroski, S. (2012). The influence of parameter fitting methods on model structure selection in automated modeling of aquatic ecosystems. *Ecological Modelling*, *245*, 136–165. 7th European Conference on Ecological Modelling (ECEM).

Cohen, S. D. & Hindmarsh, A. C. (1996). Cvode, a stiff/nonstiff ode solver in c. *Journal of Computational Physics*, *10*(2), 138–143.

Cortes, C. & Vapnik, V. N. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Craven, M. W. (1996). *Extracting comprehensible models from trained neural networks* (Doctoral dissertation, University of Wisconsin – Madison, Wisconsin, USA).

Cutler, A. & Breiman, L. (2001). Raft (random forest tool). Retrieved from www.stat.berkeley.edu/~breiman/RandomForests/cc_graphics.htm

Cutler, D. R., Edwards, T. C., Beard, K. H., Cutler, A., Hess, K. T., Gibson, J., & Lawler, J. J. (2007). Random forests for classification in ecology. *Ecology*, *88*(11), 2783–2792.

Daniels, B. C. & Nemenman, I. (2015). Automated adaptive inference of phenomenological dynamical models. *Nature communications*, *6*(8133).

Dasarathy, B. V. & Sheela, B. V. (1979). A composite classifier system design: concepts and methodology. *Proceedings of the IEEE*, *67*(5), 708–713.

Demšar, J. (2006, December). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Dietterich, T. G. (2000a). An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Machine Learning*, *40*(2), 139–157.

Dietterich, T. G. (2000b). Ensemble methods in machine learning. In *Proceedings of the 1st International Workshop on Multiple Classifier Systems* (pp. 1–15). Springer-Verlag.

Dietzel, A., Mieleitner, J., Kardaetz, S., & Reichert, P. (2013). Effects of changes in the driving forces on water quality and plankton dynamics in three swiss lakes – long-term simulations with BELAMO. *Freshwater Biology*, *58*(1), 10–35.

Domingos, P. (1998). Knowledge discovery via multiple models. *Intelligent Data Analysis*, *2*(1-4), 187–202.

Domingos, P. (2000). A unified bias-variance decomposition and its applications. In *Proceedings of the 17th International Conference on Machine Learning, ICML'00* (pp. 231–238).

Domingos, P. (2012). A few useful things to know about machine learning. *ACM Communications*, *55*(10), 78–87.

Domingos, P. & Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, *29*(2), 103–130.

Drucker, H. (1997). Improving regressors using boosting techniques. In *Proceedings of the 14th international conference on machine learning, ICML '97* (pp. 107–115). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Dunn, O. J. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, *56*(293), 52–64.

Durillo, J. J. & Nebro, A. J. (2011). jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software*, *42*, 760–771.

Džeroski, S., Panov, P., & Ženko, B. (2009). Ensemble methods in machine learning. In S. N. York (Ed.), *Encyclopedia of complexity and systems science* (pp. 5317–5325). John Wiley and Sons.

Džeroski, S. & Todorovski, L. (1993). Discovering dynamics. In *Proceedings of 10th International Conference on Machine Learning, ICML '93* (pp. 97–103). Morgan Kaufmann, San Mateo, CA.

Džeroski, S. & Todorovski, L. (1995). Discovering dynamics: from inductive logic programming to machine discovery. *Journal of Intelligent Information Systems*, *4*(1), 89–108.

Džeroski, S. & Todorovski, L. (2002). Encoding and using domain knowledge on population dynamics for equation discovery. In L. Magnani, N. Nersessian, & C. Pizzi (Eds.), *Logical and computational aspects of model-based reasoning* (Vol. 25, pp. 227–247). Applied Logic Series. Springer Netherlands.

Džeroski, S. & Todorovski, L. (2003). Learning population dynamics models from data an domain knowledge. *Ecological Modelling*, *170*, 129–140.

Džeroski, S. & Todorovski, L. (2010). Modeling the dynamics of biological networks from time course data. In S. Choi (Ed.), *Systems biology of signaling networks* (pp. 275–295). Springer Verlag.

Falkenhainer, B. C. & Forbus, K. D. (1991). Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, *51*(1-3), 95–143.

Falkenhainer, B. C. & Michalski, R. S. (1986). Integrating quantitative and qualitative discovery: The ABACUS system. *Machine Learning*, *1*(4), 367–401.

Farquhar, A. (1993). *Automated modeling of physical systems in the presence of incomplete knowledge* (Doctoral dissertation, University of Texas at Austin, Austin, TX, USA).

Ferri-Ramírez, C., Flach, P. A., & Hernandez-Orallo, J. (2002). Learning decision trees using the area under the ROC curves. In *Proceedings of the 19th International Conference on Machine Learning, ICML'02* (pp. 139–146).

Flach, P. (2012). *Machine learning: The art and science of algorithms that make sense of data.* Cambridge University Press.

Fodor, I. (2002). *A survey of dimension reduction techniques.* Lawrence Livermore National Lab., CA (US).

Forbus, K. D. (1984). Quality process theory. *Artificial Intelligence*, *24*(1-3), 85–168.

Freund, Y. (1999). An adaptive version of the boost by majority algorithm. In *Proceedings of the 12th annual conference on Computational learning theory, COLT '99* (pp. 102–113). New York, NY, USA: ACM.

Freund, Y. & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*(1), 119–139.

Freund, Y. & Schapire, R. E. (1999). A short introduction to boosting. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence, IJCAI '99* (pp. 1401–1406). Morgan Kaufmann.

Friedman, J. H., Hastie, T., & Tibshirani, R. (1998). Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, *28*, 2000.

Friedman, M. (1940). A Comparison of Alternative Tests of Significance for the Problem of m Rankings. *The Annals of Mathematical Statistics*, *11*(1), 86–92.

Gama, J. & Brazdil, P. (2000). Cascade generalization. *Machine Learning*, *41*(3), 315–343.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, *4*(1), 1–58.

Gershenfeld, N. A. (1999). *The nature of mathematical modeling.* Cambridge University Press.

Giacinto, G. & Roli, F. (2000). Proceedings of Multiple Classifier Systems: First International Workshop, MCS 2000. (Chap. Dynamic Classifier Selection, pp. 177–189). Springer.

Gilboa, Y., Friedler, E., & Gal, G. (2009). Adapting empirical equations to lake kinneret data by using three calibration methods. *Ecological Modelling*, *220*(23), 3291–3300.

Gray, G. J., Murray-Smith, D. J., Li, Y., Sharman, K. C., & Weinbrenner, T. (1998). Nonlinear model structure identification using genetic programming. *Control Engineering Practice*, *6*(11), 1341–1352.

Guyon, I., Saffari, A., Dror, G., & Cawley, G. (2010). Model selection: beyond the bayesian/frequentist divide. *Journal of Machine Learning Research*, *11*, 61–87.

Hansen, L. K. & Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(10), 993–1001.

Hastie, T. & Tibshirani, R. (1990). *Generalized additive models*. Chapman & Hall.

Hastie, T., Tibshirani, R., & Friedman, J. H. (2003). *The elements of statistical learning*. Springer.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *20*(8), 832–844.

Ho, T. K. (2000). Complexity of classification problems and comparative advantages of combined classifiers. In *Multiple classifier systems* (Vol. 1857, pp. 97–106). Lecture Notes in Computer Science. Springer.

Huynh-Thu, V. A., Irrthum, A., Wehenkel, L., & Geurts, P. (2010). Inferring regulatory networks from expression data using tree-based methods. *PLoS ONE*, *5*(9), 1–10.

Ianakiev, K. & Govindaraju, V. (2000). Architecture for classifier combination using entropy measures. In *Multiple classifier systems* (pp. 340–350). Springer.

Iman, R. L. & Davenport, J. M. (1980). Approximations of the critical region of the Friedman statistic. *Communications in Statistics - Theory and Methods*, *9*(6), 571–595.

Jolliffe, I. (2006). *Principal component analysis*. Springer Series in Statistics. Springer New York.

Jørgensen, S. E. & Bendoricchio, G. (2001). *Fundamentals of ecological modelling*. Elsevier.

Joshi, M. V., Agarwal, R. C., & Kumar, V. (2002). Predicting rare classes: Can boosting make any weak learner strong? In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '02* (pp. 297–306). New York, NY, USA: ACM.

Kalnay, E. (2003). *Atmospheric modeling, data assimilation and predictability*. Cambridge University Press.

Kaufman, L. & Rousseeuw, P. J. (1990). *Finding groups in data: an introduction to cluster analysis*. Wiley Series in Probability and Statistics. Wiley.

Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2013). Tree ensembles for predicting structured outputs. *Pattern Recognition*, *46*(3), 817–833.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI'95* (Vol. 2, pp. 1137–1143). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Kokar, M. M. (1986). Determining arguments of invariant functional descriptions. *Machine Learning*, *1*(4), 403–422.

Kong, E. B. & Dietterich, T. G. (1995). Error-correcting output coding corrects bias and variance. In *Proceedings of the 12th International Conference on Machine Learning, ICML '95* (pp. 313–321).

Kononenko, I. & Kovačic, M. (1992). Learning as optimization: stochastic generation of multiple knowledge. In *Proceedings of the 9th International Workshop on Machine Learning, ML '92* (pp. 257–262). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection*. Bradford.

Koza, J. R. (1994). *Genetic programming II: automatic discovery of reusable programs*. MIT Press.

Križman, V., Džeroski, S., & Kompare, B. (1995). Discovering dynamics from measured data. *Electrotechnical Review*, *62*(3-4), 191–198.

Kuipers, B. (1994). *Qualitative reasoning: modeling and simulation with incomplete knowledge*. MA: MIT Press.

Kuncheva, L. I. (2002). Switching between selection and fusion in combining classifiers: an experiment. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 32*(2), 146–156.

Kuncheva, L. I. (2014). *Combining pattern classifers: methods and algorithms* (2nd ed.). Wiley-Interscience.

Kuncheva, L. I. & Whitaker, C. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, *51*(2), 181–207.

Kwok, S. W. & Carter, C. (1988). Multiple decision trees. In *Proceedings of the 4th Annual Conference on Uncertainty in Artificial Intelligence, UAI '88* (pp. 327–338).

Langley, P. W. (1996). *Elements of machine learning.* Machine Learning Series. Morgan Kaufmann.

Langley, P. W., Shiran, O., Shrager, J., Todorovski, L., & Pohorille, A. (2004). Inducing explanatory process models from biological time series. In *Proceedings of the 9th Workshop on Intelligent Data Analysis and Data Mining, IDA'04* (pp. 85–90).

Langley, P. W., Simon, H. A., Bradshaw, G., & Zytkow, J. M. (1987). *Scientific discovery: Computational explorations of the creative processes.* MA: The MIT Press.

Lavrač, N. & Džeroski, S. (1994). *Inductive logic programming: techniques and applications.* Ellis Horwood Series in Artificial Intelligence. Prentice Hall.

Lavrač, N., Kavšek, B., Flach, P., & Todorovski, L. (2004). Subgroup discovery with CN2-SD. *Journal of Machine Learning Research*, *5*, 153–188.

Levatić, J., Ceci, M., Kocev, D., & Džeroski, S. (2015). New Frontiers in Mining Complex Patterns: Third International Workshop, NFMCP 2014. In A. Appice, M. Ceci, C. Loglisci, G. Manco, E. Masciari, & W. Z. Ras (Eds.), (Chap. Semi-supervised Learning for Multi-target Regression, pp. 3–18). Cham: Springer International Publishing.

Ljung, L. (1999). *System identification - theory for the user.* Prentice-Hall.

Luenberger, D. (1979). *Introduction to dynamic systems: theory, models, and applications.* Wiley.

Maclin, R. & Opitz, D. (1999). Popular ensemble methods: An empirical study. *Journal of A Intelligence Research*, *11*, 169–198.

Mason, L., Bartlett, P. L., & Baxter, J. (2000). Improved generalization through explicit optimization of margins. *Machine Learning*, *38*(3), 243–255.

McCarthy, J., Minsky, M., Rochester, N., & Shannon, C. (1955). A proposal for the Dartmouth summer research project on artificial intelligence. Retrieved from http://www-formal.stanford.edu/jmc/history/dartmouth/dartmouth.html

Melis, G. (2014). Higgs boson machine learning challenge. Retrieved from https://www.kaggle.com/c/higgs-boson

Merz, C. J. (1999). Using correspondence analysis to combine classifiers. *Machine Learning*, *36*(1), 33–58.

Mirchev, M., Duane, G. S., Tang, W. K., & Kocarev, L. (2012). Improved modeling by coupling imperfect models. *Communications in Nonlinear Science and Numerical Simulation*, *17*(7), 2741–2751.

Mitchell, T. (1997). *Machine learning.* McGraw Hill.

Nedellec, C., Rouveirol, C., Ade, H., Bergadano, F., & Tausend, B. (1996). Automatic computational discovery of chemical reaction networks using genetic programming. In L. D. Raedt (Ed.), *Advances in inductive logic programming* (pp. 82–103). Amsterdam, The Netherlands: IOS Press Amsterdam.

Nemenyi, P. B. (1963). *Distribution-free multiple comparisons* (Doctoral dissertation, Princeton University).

Okun, O., Valentini, G., & Re, M. (2011). *Ensembles in machine learning applications.* Studies in Computational Intelligence. Springer.

Opitz, D. & Shavlik, J. (1996). Actively searching for an effective neural-network ensemble. *Connection Science*, *8*(3), 337–353.

Panov, P. & Džeroski, S. (2007). Combining bagging and random subspaces to create better ensembles. In M. R. Berthold, J. Shawe-Taylor, & N. Lavrač (Eds.), *Advances in Intelligent Data Analysis VII* (Vol. 4723, pp. 118–129). Lecture Notes in Computer Science. Springer.

Pečkov, A., Džeroski, S., & Todorovski, L. (2008). Advances in Knowledge Discovery and Data Mining: 12th Pacific-Asia Conference, PAKDD 2008. In T. Washio, E. Suzuki, K. M. Ting, & A. Inokuchi (Eds.), (Chap. A Minimal Description Length Scheme for Polynomial Regression, pp. 284–295). Springer.

Perrone, M. P. (1993). *Improving regression estimation: averaging methods for variance reduction with extensions to general convex measure optimization* (Doctoral dissertation, Brown University, Providence, RI, USA).

Perrone, M. P. & Cooper, L. N. (1993). When networks disagree: Ensemble methods for hybrid neural networks. In *Neural networks for speech and image processing* (pp. 126–142). Chapman and Hall.

Polikar, R., Topalis, A., Parikh, D., Green, D., Frymiare, J., Kounios, J., & Clark, C. M. (2008). An ensemble based data fusion approach for early diagnosis of alzheimer's disease. *Information Fusion*, *9*(1), 83–95.

Robnik-Šikonja, M. & Kononenko, I. (2008). Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, *20*(5), 589–600.

Rodríguez, J. J., Kuncheva, L. I., & Alonso, C. J. (2006). Rotation forest: a new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(10), 1619–1630.

Rokach, L. (2008). Genetic algorithm-based feature set partitioning for classification problems. *Pattern Recognition*, *41*(5), 1676–1700.

Rokach, L. (2009). Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography. *Computational Statistics & Data Analysis*, *53*(12), 4046–4072.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, *33*(1-2), 1–39.

Schaffer, C. (1993). Bivariate scientific function finding in a sampled, real-data testbed. *Machine Learning*, *12*(1), 167–183.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*(2), 197–227.

Schapire, R. E. & Freund, Y. (2012). *Boosting: Foundations and Algorithms*. Adaptive computation and machine learning. MIT Press.

Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1997). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*(5), 322–330.

Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., & Džeroski, S. (2010). Predicting gene function using hierarchical multi-label decision tree ensembles. *BMC Bioinformatics*, *11*(1), 1–14.

Schmidt, M. & Lipson, H. (2009). Distilling free-form natural laws from experimental data. *Science*, *324*(5923), 81–85.

Schmidt, M., Vallabhajosyula, R. R., Jenkins, J. W., Hood, J. E., Soni, A. S., Wikswo, J. P., & Lipson, H. (2011). Automated refinement and inference of analytical models for metabolic networks. *Physical Biology*, *8*(5), 055011.

Sebestyen, G. S. (1962). *Decision-making processes in pattern recognition*. ACM monograph series. Macmillan.

Seni, G. & Elder, J. F. (2009). *Ensemble methods in data mining : improving accuracy through combining predictions*. Morgan and Claypool Publishers.

Sharkey, A. J. C. (2002). Proceedings of the 3rd International Workshop of Multiple Classifier Systems, MCS '02. In F. Roli & J. Kittler (Eds.), (Chap. Types of Multinet Systems, pp. 108–117). Springer.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2014). Constructing a library of domain knowledge for process - based modeling of neurons using the hodgkin-huxley formalism. In *Abstract Collection: 1st Human Brain Project School* (p. 25).

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015a). Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems. *Ecological Modelling*, *306*, 305–317.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015b). Predicting long-term population dynamics with bagging and boosting of process-based models. *Expert Systems with Applications*, *42*(22), 8484–8496.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2016). Modeling dynamic systems with efficient ensembles of process-based models. *PLoS ONE*, *11*(4), 1–27.

Škerjanec, M., Atanasova, N., Čerepnalkoski, D., Džeroski, S., & Kompare, B. (2014). Development of a knowledge library for automated watershed modeling. *Environmental Modelling & Software*, *54*, 60–72.

Steidl, C., Lee, T., Shah, S. P., Farinha, P., Han, G., Nayar, T., ... Gascoyne, R. D. (2010). Tumor-associated macrophages and survival in classic hodgkin's lymphoma. *New England Journal of Medicine*, *362*(10), 875–885.

Storn, R. & Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, *11*(4), 341–359.

Strogatz, S. (1994). *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. Advanced book program. Westview Press.

Štrumbelj, E. & Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, *11*, 1–18.

Štrumbelj, E. & Kononenko, I. (2011). Adaptive and Natural Computing Algorithms: Proceedings of 10th International Conference, ICANNGA 2011. In A. Dobnikar, U. Lotrič, & B. Šter (Eds.), (Chap. A General Method for Visualizing and Explaining Black-Box Regression Models, pp. 21–30). Springer.

Tanevski, J., Todorovski, L., & Džeroski, S. (2016). Learning stochastic process-based models of dynamical systems from knowledge and data. *BMC Systems Biology*, *10*(1), 30.

Tanevski, J., Todorovski, L., Kalaidzidis, Y., & Džeroski, S. (2015). Domain-specific model selection for structural identification of the Rab5-Rab7 dynamics in endocytosis. *BMC Systems Biology*, *9*(1), 1–17.

Taškova, K., Korošec, P., Šilc, J., Todorovski, L., & Džeroski, S. (2011). Parameter estimation with bio-inspired meta-heuristic optimization: modeling the dynamics of endocytosis. *BMC Systems Biology*, *5*(1), 1–26.

Taškova, K., Šilc, J., Atanasova, N., & Džeroski, S. (2012). Parameter estimation in a nonlinear dynamic model of an aquatic ecosystem with meta-heuristic optimization. *Ecological Modelling*, *226*, 36–61.

Todorovski, L. (2003). *Using domain knowledge for automated modeling of dynamic systems with equation discovery* (Doctoral dissertation, Faculty for Computer Science and Information Science, University of Ljubljana, Ljubljana, Slovenia).

Todorovski, L., Bridewell, W., Shiran, O., & Langley, P. W. (2005). Inducing hierarchical process models in dynamic domains. In K. S. Veloso M.M. (Ed.), *Proceedings of the*

20th National Conference on Artificial Intelligence, NCAI '05 (pp. 892–897). AAAI Press, Pittsburgh.

Todorovski, L. & Džeroski, S. (1997). Declarative bias in equation discovery. In F. D.H. (Ed.), *Proceedings of the 14th International Conference on Machine Learning, ICML '97* (pp. 376–384). Morgan Kaufmann, San Mateo.

Todorovski, L. & Džeroski, S. (2007). Integrating domain knowledge in equation discovery. In *Computational discovery of scientific knowledge* (Vol. 4660, pp. 69–97). Lecture Notes in Computer Science. Springer.

Tsoumakas, G., Katakis, I., & Vlahavas, I. (2004). Proceedings of the 15th European Conference on Machine Learning, ECML '04. In J.-F. Boulicaut, F. Esposito, F. Giannotti, & D. Pedreschi (Eds.), (Chap. Effective Voting of Heterogeneous Classifiers, pp. 465–476). Springer.

Ueda, N. & Nakano, R. (1996). Generalization error of ensemble estimators. In *IEEE International Conference on Neural Networks* (Vol. 1, pp. 90–95).

Valentini, G. (2003). *Ensemble methods based on bias-variance analysis* (Doctoral dissertation, Universita' di Genova, Genova, Italy).

Valentini, G. & Masulli, F. (2002). Ensembles of learning machines. In *Proceedings of the 13th Italian Workshop on Neural Nets-Revised Papers* (pp. 3–22). Wirn Vietri. London, UK, UK: Springer-Verlag.

Van Assche, A. (2008). *Improving the applicability of ensemble methods in data mining* (Doctoral dissertation, Department of Computer Science, Katholieke Universiteit Leuven, Leuven, Belgium).

Van den Berge, L. A., Selten, F. M., Wiegerinck, W., & Duane, G. S. (2011). A multi-model ensemble method that combines imperfect models through learning. *Earth System Dynamics*, *2*(1), 161–177.

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley-Interscience.

Washio, T. & Motoda, H. (1997). Discovering admissible models of complex systems based on scale-types and identity constraints. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence, IJCAI '97* (Vol. 2, pp. 810–819).

Whigham, P. & Recknagel, F. (2001). Predicting chlorophyll-a in freshwater lakes by hybridising process-based models and genetic algorithms. *Ecological Modelling*, *146*(1–3), 243–251.

Williams, R. J. & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, *1*(2), 270–280.

Witten, I. H. & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*, 241–259.

Zembowicz, R. & Zytkow, J. M. (1992). Discovery of equations: Experimental evaluation of convergence. In *AAAI* (pp. 70–75).

Zuev, Y. (1986). A probability model of a committee of classifiers. *USSR Computational Mathematics and Mathematical Physics*, *26*(1), 170–179.

Bibliography

Publications Related to the Thesis

Journal Articles

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015a). Learning ensembles of population dynamics models and their application to modelling aquatic ecosystems. *Ecological Modelling*, *306*, 305–317.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015b). Predicting long-term population dynamics with bagging and boosting of process-based models. *Expert Systems with Applications*, *42*(22), 8484–8496.

Simidjievski, N., Todorovski, L., & Džeroski, S. (2016). Modeling dynamic systems with efficient ensembles of process-based models. *PLoS ONE*, *11*(4), 1–27.

Conference Papers

Simidjievski, N., Panov, P., Todorovski, L., & Džeroski, S. (2013). Learning ensembles models of population dynamics and its application to modeling aquatic ecosystems. In *Book of abstracts : Ecological Modelling for Ecosystem Sustainability, ISEM '13* (p. 141).

Simidjievski, N., Todorovski, L., & Džeroski, S. (2013). Learning bagged models of dynamic sytstems. In *Proceedings of the 5th Jožef Stefan International Postgraduate School Students' Conference, IPSSC '13* (pp. 177–189). Best ICT paper award.

Other Publications

Conference Papers

Tanevski, J., Simidjievski, N., & Džeroski, S. (2012). Biocircuit design with equation discovery. In *Proceedings: Workshop on Learning and Discovery in Symbolic Systems Biology, in collaboration with the ECML-PKDD, LDSSB'12* (pp. 2–16).

Vidmar, N., Simidjievski, N., & Džeroski, S. (2014a). Predictive process-based modeling. In *Intelligent systems : Proceedings of the 17th International Multiconference Information Society, IS '14* (pp. 97–101).

Vidmar, N., Simidjievski, N., & Džeroski, S. (2014b). Predictive process-based modeling of aquatic ecosystem. In *Book of abstracts of the 17th Conference of Discovery Science, DS '14*.

Conference Abstracts

Simidjievski, N., Todorovski, L., & Džeroski, S. (2014a). Automated modeling of system dynamics and its applications. In *Abstract Collection: 1st Human Brain Project Educational Workshop, New Frontiers in Neuroscience and Methods of Transdisciplinary Education* (p. 10).

Simidjievski, N., Todorovski, L., & Džeroski, S. (2014b). Constructing a library of domain knowledge for process - based modeling of neurons using the hodgkin-huxley formalism. In *Abstract Collection: 1st Human Brain Project School* (p. 25).

Simidjievski, N., Todorovski, L., & Džeroski, S. (2015c). A library of domain knowledge for process-based modeling of neurons. In *Book of abstract: 10th CFGBC Symposium with ISBE and CASyM workshops From functional genomics to systems biology and systems medicine & Hands-on tutorial systems biology/medicine* (p. 32).

Biography

The author of this thesis was born 28 September 1986 in Skopje, Macedonia. He finished his primary and secondary education in Skopje, Macedonia. In 2005, he started his studies at the Faculty of Electrical Engineering and Information Technologies, University “Ss. Cyril and Methodius” in Skopje, Macedonia, focusing on Informatics and Computer Engineering. After successfully finishing his undergraduate studies (GPA 9.25) in 2009, he enrolled in a graduate program in Computer Networks and e-Technologies at the Faculty of Electrical Engineering and Information Technologies, University “Ss. Cyril and Methodius” in Skopje, Macedonia. In 2011, he successfully completed his master studies (GPA 10.00) with thesis titled “Security mechanism for wireless ad hoc networks” under the supervision of Acad. Prof. Dr. Ljupčo Kocarev.

During his undergraduate and graduate studies he held a state scholarship for talented students (awarded by the Ministry of Education and Science of the Republic of Macedonia). In the period 2009-2010, he worked as a young research assistant at the Macedonian Academy for Sciences and Arts under the supervision of Acad. Prof. Dr. Ljupčo Kocarev. In the period 2010-2011 he worked as a visiting researcher at LIFC (Laboratoire de Informatique de l’Universite de Franche-Comte) at the University of Franche-Comte, France, under the supervision of Prof. Dr. Juilien Borgues.

In 2011, he enrolled in the PhD program entitled “Information and Communication Technologies” at the Jožef Stefan International Postgraduate School, Ljubljana, Slovenia under the supervision of Prof. Dr. Sašo Džeroski and co-supervision of Prof. Dr. Ljupčo Todorovski. He collaborated with the Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia, within the EU funded FP7 project SUMO (Supemodeling by combining imperfect models), and currently collaborates on MAESTRA (Learning from Massive, Incompletely annotated, and Structured Data) and HBP (Human Brain Project). Simidjievski also holds a young researcher scholarship for doctoral studies awarded by the Slovenian Research Agency.

His research interest is in the area of machine learning, more specifically in the area of computational scientific discovery and equation discovery. In particular, he works on the development of methodology for learning ensembles of process-based models. He has published several scientific papers and has presented his work at several international conferences and workshops, both in the area of machine learning and the areas of application.