Reducing Overfitting in Process Model Induction

Will Bridewell WILLB@CSLI.STANFORD.EDU
Narges Bani Asadi NARGESB@STANFORD.EDU
Pat Langley LANGLEY@CSLI.STANFORD.EDU
Computational Learning Laboratory, Center for the Study of Language and Information, Stanford University, Stanford, CA 94305 USA

Ljupčo Todorovski LJUPCO.TODOROVSKI@IJS.SI
Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia

Abstract

In this paper, we review the paradigm of inductive process modeling, which uses background knowledge about possible component processes to construct quantitative models of dynamical systems. We note that previous methods for this task tend to overfit the training data, which suggests ensemble learning as a likely response. However, such techniques combine models in ways that reduce comprehensibility, making their output much less accessible to domain scientists. As an alternative, we introduce a new approach that induces a set of process models from different samples of the training data and uses them to guide a final search through the space of model structures. Experiments with synthetic and natural data suggest this method reduces error and decreases the chance of including unnecessary processes in the model. We conclude by discussing related work and suggesting directions for additional research.

1. Introduction

The problem of overfitting, which results in models with incorrect structure and poor generalizability, pervades machine learning. Methods such as pruning and bagging help alleviate this problem, but there has been little work on applying these methods to the development of comprehensible, explanatory models of dynamical systems. Due to differences in the representation of models, the requirements of the task, and the nature of time-series data, direct application of such methods may not be possible. In particular, neither bagging nor pruning guarantee a comprehensible final model that corresponds to established knowledge. Our research addresses the problem of overfitting in the context of scientific models that explain observations in terms of available domain knowledge.

Explanatory models take different forms depending on their use. For instance, models that characterize change over time may be represented as differential equations, which provide a powerful, though semantically poor, language. Recent work has examined methods for embedding the terms of such equations in quantitative processes (Langley et al., 2002; Langley et al., 2003; Todorovski et al., in press). For example, consider the equations from the well-known Lotka–Volterra model of predation,

$$\begin{aligned} \dot{F} &= \gamma F - \alpha F C \\ \dot{C} &= \epsilon \alpha F C - \delta C \ , \end{aligned}$$

in which C denotes the population of the predator and F represents the population of the prey. A quantitative process model would include the term γF in a process that corresponds to prey growth, with γ identified as a parameter and F as a variable of appropriate type. The term $-\delta C$ would be embedded in a separate process for predator loss, and the last two terms, $\epsilon \alpha F C$ and $\alpha F C$, would be grouped into a single process since, together, they define the predatory interaction between the two species.

Research on quantitative process models addresses questions in representation, utilization, induction, and revision, with an emphasis on identifying plausible, accurate, and understandable models. However, current techniques for inducing such models can lead to overfitting the training data. Ensemble methods such as

Appears in *Proceedings of the Twenty-Second International Conference on Machine Learning*, Bonn, Germany, 2005.

Table 1. A hierarchical process model for a simple aquatic ecosystem in which a plankton species (*phyto*) depends on the nutrients *nitrate* and *phos*.

process primary_conc_change(*phyto*,{*nitrate*, *phos*})
  process limited_growth(*phyto*, {*nitrate*, *phos*})
    equations $d[phyto.conc, t, 1] = 2.3 * phyto.limit\_rate$
    process holling_type_1(*nitrate*, *phyto*)
      equations $phyto.limit\_rate = nitrate.conc$
    process holling_type_2(*phos*, *phyto*)
      equations
        $phyto.limit\_rate = phos.conc/(phos.conc + 5.2)$
  process exponential_loss(*phyto*)
    equations $d[phyto.conc, t, 1] = -1.2 * phyto.conc$

bagging (Breiman, 1996) combat this problem when building classifiers, and it seems reasonable that modifications of these methods could provide a solution that works well with process models. Nevertheless, such a solution must respect the constraints of the process modeling task—that is, it must both produce a comprehensible model and work well with data from nonlinear dynamical systems. The first constraint requires a method for merging the multiple models produced by an ensemble learner, while the second necessitates a specialized means for generating training samples from the original data set.

In the pages that follow, we present a system called FUSE that adapts bagging to process model induction so that it produces a single, comprehensible explanation.[1] In the next section, we review the process modeling paradigm, including an approach to inducing hierarchical process models from time-series data and background knowledge. After this, we discuss a technique for generating multiple training sets and present a method for building a single model from an ensemble of learned predictors. We then report experiments that demonstrate the existence of an overfitting problem in process model induction, along with evidence that our new method mitigates this effect on both synthetic and natural data. Finally, we discuss related research on equation discovery and ensemble learning, after which we suggest directions for additional work on inducing accurate and comprehensible models.

2. Inductive Process Modeling

Scientific explanations often take the form of models wherein domain specific processes affect some entities under observation. One class of explanations represents these processes in terms of algebraic and differential equations that relate the properties of entities within a complex system. Such processes may also include conditions for their activation that depend on the state of the entities. Langley et al. (2002) refer to this type of explanation as a *quantitative process model*, to distinguish it from Forbus' (1984) earlier notion of a qualitative process model.

Table 1 shows a process model for an ecosystem with a primary producer that feeds on two nutrients in the environment. As in other recent work (Todorovski et al., in press), this model is organized as a hierarchy, in that each process decomposes into a set of one or more subprocesses until it reaches leaf nodes that specify primitive processes. Also, system variables, which need not be observable, are associated with their respective entities, so that, for instance, phyto.conc refers to the concentration of phytoplankton.

The top-level process in Table 1, primary_conc_change, relates the population of phytoplankton to the nutrients nitrate and phosphate, and it consists of subprocesses that characterize both the limited growth and the exponential loss of the primary producer. In this model, the process limited_growth has one equation that relates the concentration of phytoplankton to its growth limitation rate; that is, the first derivative of the phytoplankton concentration with respect to time, d[phyto.conc, t, 1], equals 2.3 times phyto.limit_rate. The two nutrients affect the limit rate as shown in the associated subprocesses. Exponential loss, which lacks subprocesses, indicates the loss function for phytoplankton within the ecosystem.

Given such a process model, we can assemble the corresponding system of differential and algebraic equations so that, with initial values for the variables, we can simulate the model's behavior over time. In some cases, a variable's behavior will be explained by interactions among processes, as with phyto.conc in limited_growth and exponential_loss. Each variable in the model states how such multiple effects should be combined; in this case, the influences are added. Once assembled, the program simulates the system of equations by repeating a two-step process, first solving the differential equations and then updating the algebraic equations for a particular time step, using only the active processes. This produces values for each observable and unobservable variable at all desired times.

Existing methods for inducing process models require background knowledge cast as generic processes that relate entity types and that define the space of possible model structures. In Table 2, primary_conc_change forms the root of a hierarchy that relates a primary_producer to a nutrient. This process decomposes into one of growth, which is required, and loss, which is

[1] FUSE = **F**orming **U**nified **S**cientific **E**xplanations.

Table 2. A simplified hierarchy of generic processes for modeling population dynamics.

entity primary_producer
  variables $conc\{sum\}$, $limit\_rate\{prod\}$
  parameters $max\_growth[0, inf]$, $loss\_rate[0, inf]$
entity nutrient
  variables $conc\{sum\}$
  parameters $toCratio[0, inf]$
process primary_conc_change
  relates $P\{primary\_producer\}$, $N\{nutrient\}$ <0 to inf>
  processes $growth(P, N)$, $[loss(P)]$
process exponential_growth$\{growth\}$
  relates $P\{primary\_producer\}$
  equations $d[P.conc, t, 1] = P.max\_growth * P.conc$
process limited_growth$\{growth\}$
  relates $P\{primary\_producer\}$, $N\{nutrient\}$ <0 to inf>
  processes limiting_factor(N, P)
  equations
    $d[N.conc, t, 1] = -1 * N.toCratio * P.max\_growth * P.limit\_rate * P.conc$
    $d[P.conc, t, 1] = P.max\_growth * P.limit\_rate * P.conc$
process holling_type_1$\{limiting_factor\}$
  relates $N\{nutrient\}$, $P\{primary\_producer\}$
  equations $P.limit\_rate = N.conc$
process holling_type_2$\{limiting_factor\}$
  relates $N\{nutrient\}$, $P\{primary\_producer\}$
  parameters $lambda[0, inf]$
  equations $P.limit\_rate = N.conc/(N.conc + lambda)$
process exponential_loss$\{loss\}$
  relates $P\{primary\_producer\}$
  equations $d[P.conc, t, 1] = -1 * P.loss\_rate * P.conc$

optional. Continuing down the hierarchy, the growth process can be elaborated by a limited or an exponential form, with the former requiring a limiting_factor for each relevant nutrient (as seen in Table 1). In addition, as with the specification of entity types, which serve as slots for actual entities, generic processes provide upper and lower bounds on parameters rather than specific values.

Apart from connecting specific process models to domain knowledge, generic processes also provide a means for inducing comprehensible explanations for a given set of time-series observations. For instance, the system HIPM (Todorovski et al., in press), on which we build, carries out a two-level, heuristic beam search through the space of instantiated models. Beginning with the empty model, the system refines all models in the current beam by adding one or more processes to each candidate, depending on the constraints imposed by the generic process hierarchy. On the first refinement step, HIPM generates all allowable model structures that exclude optional processes. On later steps, the system adds unused optional processes, one at a time, to each candidate in which they are permitted. These may in turn allow the addition of other optional elements during later iterations.

After every refinement step, each candidate model consists of primitive processes that relate particular entities but lack parameter values. Thus, the second level of search calls a nonlinear least-squares algorithm (Bunch et al., 1993) that pursues a second-order gradient descent through the parameter space guided by the sum of squared errors. A user-specified number of random restarts in this space helps the system avoid local minima. The values for one restart are based on results obtained from teacher forcing (Williams & Zipser, 1989), which estimates parameters based on their ability to predict observations at time $T+1$ from those at T. After fitting the parameters for each model structure, HIPM ranks it in the beam according to its error score, selects the N best candidates, and repeats the two steps, continuing until none of the generated alternatives has lower error than its parents. Preliminary studies with this system have produced results on ecosystem modeling tasks that are substantially better, in terms of predictive accuracy and plausibility, than earlier approaches to inductive process modeling.

3. Reducing Overfitting in Inductive Process Modeling

Despite the encouraging results obtained with HIPM on scientific data sets, evidence indicates that the system can produce models that fit the training data well but that generalize poorly to novel trajectories. Moreover, informal studies with time series generated synthetically from a known model revealed that, on some runs, the algorithm produces models with unnecessary processes, which is reminiscent of early methods for decision-tree induction that tended to incorporate idiosyncratic tests. Together, these findings indicate the need for additional measures to mitigate overfitting.

The analogy with decision-tree methods suggests pruning (Quinlan, 1993) as a promising response, since the technique has been widely used in that paradigm, but closer inspection reveals problems with this idea. Postpruning methods, which construct an overly detailed model and then remove undesirable components, are not appropriate for HIPM because they could produce candidates that fall outside the model space defined by the hierarchical generic processes. Prepruning, which involves early halting of model refinement, seems more appropriate but requires a termination criterion. Some earlier work on inductive process modeling attempted to use minimum description length for this purpose. In particular, Langley et al. (2003) reported a method

Table 3. Example processes used to create an ensemble of ten induced models and the identifiers of the models in which they occur.

Generic Process	Identifiers
primary_conc_change	all
exponential_growth	1, 2, 3
logistic_growth	4, 5, 6
limited_growth	7, 8, 9, 10
holling_type_1 (nitrate)	7, 8, 9, 10
holling_type_2 (nitrate)	—
holling_type_1 (phos)	7, 8
holling_type_2 (phos)	9, 10
exponential_loss	1, 4, 9, 10

that evaluates alternative models using the expression

$$M_d \quad = \quad M_v + M_p \cdot log(n) + n \cdot log(M_e) \ ,$$

where M_v is the number of variables, M_p is the number of parameters, n is the number of training cases, and $log(M_e)$ specifies the bits required to communicate the mean squared error. Experiments with synthetic data indicated that using this measure for model selection produced the correct candidate structure in some cases but selected overly simple models, which omitted processes, in others.

In recent years, ensemble approaches such as bagging (Breiman, 1996) have superseded pruning as general-purpose methods for improving accuracy in supervised induction. Bagging is designed specifically to mitigate overfitting by reducing the variance of learned models. The effectiveness of this technique makes it an appealing approach for incorporation into the process modeling framework. Unfortunately, it would produce combined models that predict but do not explain the data, in that they would have no clear interpretation for domain scientists. Within the process modeling framework, effective use of an ensemble approach requires an additional step that generates a single, comprehensible model from the multiple structures produced from different training sets.

In response, we have developed FUSE, which incorporates an ensemble method that is similar in spirit to bagging but differs on a number of fronts. Assuming data are independently and identically distributed, traditional bagging samples the original observations with replacement to generate multiple training sets, thereby allowing some cases to occur multiple times within a given set. However, repeated observations of this sort can cause difficulties for a system like HIPM that works with time series from nonlinear systems by distorting parts of the trajectory and causing poor estimates of model parameters. The independently and identically distributed assumption does not hold for such domains. Our approach addresses both creation of data sets and merging of the induced ensemble.

Rather than using the bootstrap (Efron & Tibshirani, 1993) as in bagging, FUSE lets the user specify the number of training sets, k, and the size of each set in terms of the percentage, n, of the initial data. The system creates the requested sets by sampling from the original data following a uniform distribution and without replacement. As a result, all observations within a given training set are unique, but the sampled cases differ across sets. An alternative to this method would involve selecting small contiguous chunks of data at random. Although this would preserve local relationships in the trajectories, it could lose key global features, especially when the measurement rate is coarse. Since the global features tend to be more relevant to our current task (i.e., building models that reproduce observed trajectories from initial conditions), we chose the first data selection method, which better matches this goal.

Once FUSE produces the k time series with this sampling method, it applies HIPM to each set to create k quantitative process models that may differ not only in their parameters but also in the processes they incorporate. To combine these alternatives into a single model, the system takes advantage of the hierarchical nature of the generic processes. Beginning at the root of the hierarchy, which is always present, and working toward the leaves, the program selects those processes in the first level that appear most often within the k models. For example, Table 3 refers to the generic processes from Table 2 and the models in which their instantiations occur. In this case, FUSE begins by placing the root process (primary_conc_change) into the final structure. Since this process requires a growth subprocess, the system compares the frequencies of the alternatives and selects limited_growth, which occurs most often. Similarly, it selects holling_type_1 as nitrate's limiting factor, since this appears in all models that contain its parent, limited_growth.

FUSE uses the frequency counts differently when deciding whether to incorporate an optional process. In particular, the system adds the subprocess only if it occurs in more than half of the models that contain its parent. For instance, the root process in the example specifies an optional loss component. As Table 3 shows, exponential_loss, which is the only process that can fill the role, occurs in four of the ten models that contain primary_conc_change. Thus, the program leaves exponential_loss out of the final structure.

Table 4. The process model structure that FUSE generates from the frequency counts in Table 3.

process primary_conc_change(*phyto*, {*nitrate*, *phos*})
  process limited_growth(*phyto*, {*nitrate*, *phos*})
    process holling_type_1(*nitrate*, *phyto*)
    process holling_type_1(*phos*, *phyto*)

In the event of a tie, the program selects the process that appears in combination most often with processes already chosen for the combined model. FUSE begins at the level of the hierarchy where the tie occurs, and works upward until it can make a choice. If this step does not break the tie, the system averages the error scores of the models associated with each alternative and selects the process with the lowest score. In Table 3, a tie exists between the holling_type_1 and holling_type_2 processes that use phosphate. FUSE first notices that the other process already included at this level, holling_type_1 using nitrate, appears with both options at the same frequency. In response, the program moves up one level in the hierarchy and notices that exponential_loss never occurs with the holling_type_1 option, but always appears with holling_type_2. Since the final structure does not contain exponential_loss, FUSE selects holling_type_1 as the final component of the model. Having determined the structure in Table 4, the system calls HIPM's parameter estimation routine with the original time series to determine the combined model's parameters.

In addition to the method just described, we considered some alternatives. One simple scheme would include all processes that occur in at least j of the k models, but this could generate model structures that contain processes known at the outset to be mutually exclusive. The clear advantage of FUSE is that it carries out search through the same AND/OR space as the HIPM system, but uses probability of inclusion in the multiple models to make choices rather than squared error. As a result, it is guaranteed to induce a single, well-formed process model that obeys the constraints HIPM uses to generate its models. Of course, whether this method works in practice to reduce overfitting is an empirical question, to which we now turn.

4. Experimental Studies

As noted earlier, overfitting involves poor generalization to novel test data, but this phenomena is often associated with learned models that contain incorrect or unnecessary components. We designed FUSE to address both issues, so we crafted experiments to examine them separately. One study used natural data to show that the combined model generalizes better to unseen cases than does a single process model induced from one training set. We then used synthetic data to show that the ensemble technique recreates the model structure used for data generation more reliably than the base HIPM algorithm.

Table 5. The sum of squared errors (SSE) and the squared correlation coefficient (r^2) for the baseline approach and FUSE on data from the Ross Sea ecosystem.

Data	Baseline		Ensemble	
	SSE	r^2	SSE	r^2
Fold 1	4505	0.74	2904	0.95
Fold 2	625	0.88	348	0.96
Fold 3	73426	—	3842	0.80
Fold 4	4312	0.78	1815	0.82
Fold 5	2102	0.86	267	0.98
Mean	16994	0.82	1835	0.90

The natural data used to evaluate the effectiveness of our approach comes from an ecological domain. The Ross Sea in Antarctica has drawn the attention of many researchers (e.g., Arrigo et al., 2003) due to its relatively simple food web, which means that many processes important in open ocean or limnological systems have minimal effects. Phytoplankton plays a central part in this ecosystem due to its role as a primary producer, and it undergoes a yearly cycle of population increase and decrease affected primarily by the amount of light, nutrients, and possibly zooplankton. Characterizing these cycles remains an open problem in ecology and presents a difficult challenge for inductive process modeling.

In this experiment, the data consisted of 188 daily measurements taken from the Ross Sea during the period when phytoplankton bloom. We performed five-fold cross validation using the standard method for creating the folds (i.e., each datum was assigned to one of the partitions that composed the folds). We applied both HIPM and FUSE to each fold, using a beam width of 32 for the structural search and 8 random restarts per structure for the parameter-optimization routine. For FUSE, we set the percentage of the initial data, n, to 80% and the number of generated data sets, k, to 10. Finally, we calculated the correlation coefficients and squared errors of the resultant models.

Table 5 shows the results of this experiment. For each fold, the ensemble approach improved the square of the correlation coefficient (r^2) and the squared error (SSE) on the test set with respect to the base method,

Table 6. Mean number of extra and missing processes in the models generated by the baseline and ensemble training methods. Lower scores indicate better results.

Noise Level	Baseline		Ensemble	
	extra	missing	extra	missing
0%	1.00	0.60	0.40	0.20
5%	1.00	0.60	0.80	0.20
10%	1.20	0.60	0.40	0.40
15%	1.40	0.80	1.60	0.80
Mean	0.92	0.65	0.80	0.40

performing more accurately. The third fold presented a strong challenge to HIPM, leading to baseline trajectories that were uncorrelated with the observations. However, removing this outlier gives squared errors of 2886 for the baseline and 1333 for the ensemble method, which does not alter the general outcome. Thus, FUSE appears to reduce overfitting in the sense of lowered error on scientific data that hold significant interest to ecologists.

Although the results with natural data demonstrate that our ensemble method can produce models that better generalize to actual scientific data, they do not show that it reduces the number of incorrect processes in the final model. To evaluate this aspect, we created synthetic data from a hand-crafted process model. This involved six primitive processes that related four variables associated with organisms, nutrients, and other entities. We simulated this model over 100 days from five different sets of initial values to produce five trajectories for each of the variables. We then added zero, five, ten, and fifteen percent Gaussian noise to these observations to generate training sets covering a wide range of situations.

For the control condition, we trained HIPM on each of the 20 data sets and counted the number of processes in the induced models that were extra or missing. For example, if the system replaced limited_growth with exponential_growth in a model, we recorded one missing and one extra process. For the ensemble condition, we invoked FUSE on the same data sets with the parameter k set to 5 and with n set to 80 percent. The system applied HIPM to each of the k subsets in turn and combined the induced models using the method described earlier. Again, we tabulated the number of variations from the true model and averaged the resulting scores within noise levels. All runs of HIPM used a beam width of 32, which led it to consider hundreds of model structures during each search.

Table 6 shows the results of this experiment. In general, FUSE produced models with fewer extra and missing processes than the baseline system. Interestingly, all of the missing processes resulted from the substitution of one process for another of the same type. This finding raises the question of whether these substitutions were more or less prevalent than the addition of unnecessary processes. The base method generated more substitutions (0.65 per model) than extra processes (0.5 per model), whereas the ensemble method produced the same number of both on average (0.4 per model) and fewer of both than the baseline. Upon closer examination of the models composing the ensembles, we noted substantial variation in certain parts of the structure, which indicates that FUSE was actively making choices and not just combining a set of identical model structures.

The results on both the Ross Sea data and the synthetic data are encouraging. We found that, on actual scientific data, the ensemble method increases the predictive accuracy of the model and, on synthetic data, it makes the identification of the generating structure more probable. We expect these findings to generalize to different configurations of the generating model and to alternative libraries of generic processes. However, we believe that varying the number of combined models and the size of the respective training sets will have a noticable effect, and we intend to carry out more systematic analyses as a means to better understand the FUSE algorithm's behavior.

5. Related Research

Although inductive process modeling is a relatively new paradigm, the approach we have described draws on a number of earlier traditions. For example, it builds directly on research in equation discovery, but differs from most work in this area (e.g., Langley, 1981; Żytkow et al., 1990; Washio et al., 2000) in its emphasis on explanatory models and in its reliance on domain knowledge. Our method comes closer to work by Bradley et al. (2001) and Todorovski (2003), which also deals with the induction of differential equation models from time series and uses background knowledge to constrain search through the hypothesis space.

We have also incorporated methods from the literature on parameter optimization and on system identification (Åström & Eykhoff, 1971), which develops methods that fit dynamical models to temporal data. However, we have embedded our parameter-estimation techniques within a higher-level search through the space of model structures, which we organize in hi-

erarchical terms. In addition, we have adapted ideas from the field of qualitative physics, including our use of qualitative processes to group equations and our approach to constructing models by composing generic model fragments (Falkenhainer & Forbus, 1991).

We have discussed these relations at greater length in earlier papers. The novel contribution here borrows from the entirely distinct literature on ensemble learning. Researchers in this area have developed a wide range of methods for combining learned models to reduce predictive errors. Clearly, our approach incorporates key ideas from bagging (Breiman, 1996), but it also differs in important ways. The most significant is that, although our method combines the results of runs from multiple data sets, the end result is a single quantitative process model.

Our motivation was to produce an explanatory account that would be comprehensible to domain scientists, rather than a traditional ensemble that is accurate at the expense of interpretability. Domingos' (1998) CMM algorithm was driven by the same concern, although it uses a bagged ensemble to generate additional data for inducing a single classifier. This technique reduces variance of the learned model but tends to increase its complexity, whereas our approach produces simpler models. However, the two methods are complementary, and we plan to combine them in our future work.

We have already noted that FUSE's ability to construct one model from a collection depends on the organization of HIPM's search. This revolves around the notion of declarative bias, which figures prominently in research on inductive logic programming (e.g., Adé et al., 1995). Todorovski (2003) has already utilized this framework to constrain his algorithms for equation discovery, but our approach goes farther by taking advantage of the hierarchical structure of process models. We hypothesize that analogous approaches to inducing comprehensible ensembles will be possible for other hierarchical frameworks.

6. Concluding Remarks

In the preceding pages, we reviewed the recent paradigm of inductive process modeling and noted existing methods' tendency to overfit the training data. This problem is not surprising, since it arises naturally in the early stages of research on any class of induction methods. In response, we turned to the notion of ensemble learning, an approach now widely used to reduce overfitting. Some version of bagging seemed a likely candidate, since we believed our method would benefit most from a reduction in variance.

However, process model induction poses a special challenge for ensemble techniques because it is centrally concerned with producing results that scientists will find comprehensible. The process modeling framework addresses this concern by encoding individual models as differential equations embedded in domain-relevant processes, but, because ensembles combine multiple models, they typically gain in accuracy at the cost of complexity and comprehensibility. We needed some way to retain the benefits of model combination while retaining interpretability.

Our response was to induce a set of quantitative process models from bootstrapped samples of the training data, as in bagging, but then to use this set to infer a single model structure. This scheme involved carrying out search through the same refinement space as the base method, using the most common decision at each choice point to determine the model structure and then estimating its parameters on the entire training set. Experiments on an ecosystem modeling task with synthetic and natural data provided evidence that this approach produces process models with lower test error and fewer missing or extra components.

Despite these encouraging results, there remain some open issues that we should address in future research. First, to establish our method's generality, we should demonstrate its ability to reduce overfitting on data sets from other scientific domains. In addition, we should carry out more experiments with synthetic data to determine how factors such as the number of combined models and the number of training cases influence the technique's behavior. We should also consider other approaches to sampling time series and explore ways to combine our ensemble method with the CMM algorithm's ability to generate additional data. Taken together, these studies should give us a fuller understanding of the conditions under which our approach supports the induction of accurate process models.

Acknowledgements

This research was supported by NSF Grant No. IIS-0326059 and by NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation. We thank Kevin Arrigo for data from and knowledge about the Ross Sea ecosystem, Oren Shiran for his contributions to the HIPM system, and Bradley Efron for his advice on applications of bootstrap sampling methods to time series.

References

Adé, H., Raedt, L. D., & Bruynooghe, M. (1995). Declarative bias for specific-to-general ILP systems. *Machine Learning*, *20*, 119–154.

Arrigo, K. R., Worthen, D. L., & Robinson, D. H. (2003). A coupled ocean–ecosystem model of the Ross Sea: 2. Iron regulation of phytoplankton taxonomic variability and primary production. *Journal of Geophysical Research*, *108*, 3231.

Åström, K. J., & Eykhoff, P. (1971). System identification—A survey. *Automatica*, *7*, 123–167.

Bradley, E., Easley, M., & Stolle, R. (2001). Reasoning about nonlinear system identification. *Artificial Intelligence*, *133*, 139–188.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*, 123–140.

Bunch, D. S., Gay, D. M., & Welsch, R. E. (1993). Algorithm 717: Subroutines for maximum likelihood and quasi-likelihood estimation of parameters in nonlinear regression models. *ACM Transactions on Mathematical Software*, *19*, 109–130.

Domingos, P. (1998). Knowledge discovery via multiple models. *Intelligent Data Analysis*, *2*, 187–202.

Efron, B., & Tibshirani, R. J. (1993). *An introduction to the bootstrap*. New York City: Chapman & Hall.

Falkenhainer, B., & Forbus, K. D. (1991). Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, *51*, 95–143.

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, *24*, 85–168.

Langley, P. (1981). Data-driven discovery of physical laws. *Cognitive Science*, *5*, 31–54.

Langley, P., George, D., Bay, S., & Saito, K. (2003). Robust induction of process models from time-series data. *Proceedings of the Twentieth International Conference on Machine Learning* (pp. 432–439). Washington, D.C.: AAAI Press.

Langley, P., Sánchez, J., Todorovski, L., & Džeroski, S. (2002). Inducing process models from continuous data. *Proceedings of the Nineteenth International Conference on Machine Learning* (pp. 347–354). Sydney: Morgan Kaufmann.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco, CA: Morgan Kaufmann.

Todorovski, L. (2003). *Using domain knowledge for automated modeling of dynamic systems with equation discovery*. Doctoral dissertation, Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia.

Todorovski, L., Bridewell, W., Shiran, O., & Langley, P. (in press). Inducing hierarchical process models in dynamic domains. *Proceedings of the Twentieth National Conference on Artificial Intelligence*. Pittsburgh, PA: AAAI Press.

Washio, T., Motoda, H., & Niwa, Y. (2000). Enhancing the plausibility of law equation discovery. *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 1127–1134). San Francisco, CA: Morgan Kaufmann.

Williams, R., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, *1*, 270–280.

Żytkow, J. M., Zhu, J., & Hussam, A. (1990). Automated discovery in a chemistry laboratory. *Proceedings of the Eighth National Conference on Artificial Intelligence* (pp. 89–894). Boston, MA: AAAI Press.