Statistical Relational Learning for Prognostics

Jonas Vlasselaer JONAS.VLASSELAER@CS.KULEUVEN.BE
Wannes Meert WANNES.MEERT@CS.KULEUVEN.BE
KU Leuven, Department of Computer Science, Celestijnenlaan 200A, 3001 Heverlee, Belgium

Keywords: Prognostics, Temporal models, Statistical Relational Learning

Abstract

The field of prognostics aims to predict the remaining useful life of a component or machine by means of probabilistic models. These models typically need to satisfy different requirements imposed by the available data, the expert knowledge and the prediction task at hand. Graphical probabilistic models are popular tools in prognostics but satisfy these requirements only partially. We introduce some examples to illustrate these requirements, analyze the most suitable graphical model and propose to use Statistical Relational Learning (SRL) for prognostics to satisfy more requirements. The expressiveness offered by this framework, however, has a computational cost and in this paper we investigate how current state-of-the-art SRL systems perform on prognostics tasks.

1. Introduction

The affordance of computing power, data storage and sensors make automated monitoring increasingly appealing for the mechatronics industry. This has lead to machines that are monitored more closely and detect signal deviations from normal behavior at a much earlier stage. But not only deviations can be observed, more complex probabilistic models also allow for prediction of future states of the machine. Where previously an expert was needed to inspect a machine and estimate how the machine would evolve over time (e.g., degradation), this can now be (partially) automated.

In this paper we focus on models for prognostics and more specifically on the prediction of the Remaining Useful Life (RUL) of a machine and its components. Determining the RUL accurately has a positive impact on the efficiency of a plant. Popular methods for prognostics are probabilistic (graphical) models like (hidden) Markov models, hidden semi-Markov models, support vector machines and Bayesian networks. Although these models are able to efficiently deal with particular temporal processes, the constraints they impose are often too limiting for a real-world scenario. Often, a hierarchical model, or a model with functional and relational dependencies between classes of objects, or a combination of different models is necessary. Furthermore, most probabilistic models require in-depth knowledge to build such models. Ideally, a domain expert would be able to express his knowledge in a declarative, modular manner without the need to understand the underlying algorithms.

Statistical Relational Learning (SRL) techniques (Getoor & Taskar, 2007; De Raedt & Kersting, 2003) extend the probabilistic model framework with relational and functional elements, allows us to express expert or background knowledge about the system, and can express structural information such as time. The more logic based techniques in the field of SRL also allow a declarative and modular way of expressing knowledge. As we are mainly interested in formalisms that are flexible enough to fulfill all requirements, we will focus on logic based SRL techniques.

In this paper we investigate whether SRL could offer a solution to some of the (artificial) restrictions we encounter in the field of prognostics when using traditional probabilistic models. As a first contribution we identify a set of requirements that are often encountered in a real-world prognostics setup. The second contribution is a comparison of how current state-of-the-art SRL systems perform on a set of models adhering to the requirements defined in this paper. We are mainly interested in the trade-off between efficiency and the gained expressiveness.

Appearing in *Proceedings of BeneLearn and PMLS 2012*.
Copyright 2011 by the author(s)/owner(s).

2. Prognostics

In this section, we introduce the field of prognostics and enumerate some specific requirements inherent to this field. These requirements are distilled from the real-life data sets and problems we encountered in the context of the Prognostics for Optimal Maintenance project.[1] This project aims to automate and optimize the maintenance strategy for industrial machinery.

2.1. Background

Maintenance is typically one of the biggest costs for large manufactures and therefore the choice of a good maintenance strategy is crucial. In the search for an optimal strategy, companies have to deal with a trade-off: if one waits too long to do maintenance, an unexpected breakdown of a component or machine can result in a loss of productiveness; a more progressive strategy, on the other hand, ensures a continuous productivity process but results often in an unnecessary replacement of components.

To find the most optimal moment for maintenance, particular information about the machine or its components is needed. This information is mostly obtained by a set of sensors which measure specific characteristics of the component at a constant rate. An expert typically searches irregularities or trends in this temporal data and decides which machine needs more attention because of the risk of a breakdown. In this way, the expert tries to estimate the Remaining Useful Life (RUL) of the system, given the sensor information up until that moment and his past experience or knowledge about the machine.

Although an expert can often make an accurate estimate of the RUL, there are several reasons why companies are interested to use machine learning techniques to (partially) automate this task. **Information management**: knowledge needs to be managed such that it can be reused once an expert leaves the company. **Scalability**: an expert is limited in time and space and therefore multiple similar experts could be needed to monitor similar machines, especially if they are spread over different locations. **Complexity**: machine learning techniques can typically handle more information or explore more complex relations between machines and components and as such improve the accuracy of the predictions. **Prognostics as a service**: besides prognostics for the manufacturing process one could think about prognostics for the products a company fabricates. In this case it is clear only automated prognostics has a real benefit.

[1] POM2 IWT-SBO project (http://www.pom2sbo.org)

To estimate the RUL of a system we first need to model its degradation and next predict the moment where this model will cross a certain threshold. Unfortunately, most components do not allow us to observe the degradation directly without shutting down the machine. For example, the real degradation of ball bearings can often only be observed if we open them up and inspect the condition of the balls and the inside of the cage. Because the real degradation is hidden, it is necessary to measure other characteristics and use them to infer knowledge about the condition of the component. This relation is mostly probabilistic because sensors occasionally make some measurement errors or they can be influenced by external, and probably unknown, events.

Prognostics is not only relevant for components and machines, similar models and techniques are useful for other applications. One of them is health care for elderly persons in service flats or retirement homes. Different sensors could be used to infer some information about the hidden health condition of the residents. The goal would be to predict a dramatic change in the health condition of a person such that medical personnel can be notified in time.

2.2. Requirements

Based on the data and tasks involved with a prognostics problem, we can formulate a set of requirements which a probabilistic model, ideally, fulfills to help solve a prognostics problem:

1. **Handle probabilistic and partial observable data.** The real degradation is typically unobservable and its relation with the measured data is often probabilistic.
2. **Efficient inference and learning for temporal data.** Multiple sensor measurements per day over a period of several weeks or months accumulates into a large temporal dataset. More observations lead in general to a better estimation of the RUL and therefore efficient inference is necessary. Because the parameters in the model are ideally learnable form observation sequences, efficient learning is needed.
3. **Continuous distributions to model duration.** To plan maintenance in the most optimal way, an estimation of the time until failure is needed. This can be, for example, a (mixture of) Gaussian or Gamma distribution.
4. **Multiple observations for every time instance, possibly related.** Different sensors can be used to measure several characteristics of a

component and this results in multiple observations per time instance. The existing relations between these observations are ideally included in the model to improve accuracy.

5. **Structural and modular representation of how a system evolves over time.** A structural and modular representation allows the expert to represent his knowledge in a more abstract way, independent from all the details of the system. Additionally, it allows to easily reuse or transfer parts of the model to other similar systems.
6. **A unifying framework to represent knowledge about different types of interactions.** A domain expert should be able to write down his knowledge about a system without the need to be an expert in probabilistic modeling or algorithms.

3. Temporal model approaches

In this section, we introduce some examples of increasing complexity to illustrate the problems encountered in prognostics. We relate these problems to the requirements defined above and briefly analyze the most suitable temporal model.

3.1. Hidden Markov models

The first problem is to model the degradation of a ball bearing given a single sensor measurement for each time point. The real degradation is hidden but is assumed to have an influence on the vibration amplitude measured at the outside of the bearing. This can be modeled with a Hidden Markov Model (HMM) (Rabiner, 1989) which represents the degradation as a finite set of hidden states related according to a certain probability distribution. As shown in figure 1(a), the degradation (D) is a cause for the sensor to measure a certain vibration amplitude (V). A HMM assumes the underlying process to be memoryless and therefore allows efficient inference and parameter learning for long sequences of data.

3.2. Hidden-semi Markov models

A HMM can be used to model the ball bearing in our first example, and satisfies as such the first two requirements defined in section 2.2, but is too limited to model the RUL of a component. Assume the degradation can be modeled with two hidden states, the RUL is equivalent to the time the model stays in the G(ood) state and makes a transition towards the B(ad) state. This is by definition an exponential distribution (Rabiner, 1989) and therefore limits the capability of an HMM to model the real distribution over the RUL.

A Hidden semi-Markov Model (HsMM) allows the underlying process to be a semi-Markov chain and explicitly defines the duration or sejourn time for every state (Yu, 2010). This duration is related to the number of observations produced while in the state, as shown in figure 1(b), and therefore the self-transition probabilities are equal to zero. A HsMM allows the state duration to be modeled by, for example, a Gaussian or Gamma distribution and satisfies as such the third requirement. The duration specified for the G(ood) state can be the lifetime of the ball bearing, specified by the producer with a certain mean and variance.

3.3. Dynamic Bayesian networks

Until now, we assumed to have only one observation for every time instance but in general also more observations could be available. For example, the vibration caused by a ball bearing typically consist of two orthogonal components which can be measured separately with two different sensors. Because it is cheaper to use only one sensor and measure the composed signal, the separate components are only measured if the composed signal reaches a threshold and the expert decides to inspect the machine with two separate sensors. This results in a partial observed dataset where the relations between the signals, as shown in 1(c), are incorporated in the model.

A Dynamic Bayesian Network (DBN) (Mihajlovic & Petkovic, 2001) is a generalization of a HMM used to model more complex temporal data. Besides the relations between multiple observations, a DBN allows to model more complex relations between the hidden states and satisfies as such the fourth requirement defined in section 2.2. Note that the term "dynamic" implies that a dynamic system is modeled and does not mean the structure of the model changes over time.

3.4. Logical hidden Markov models

The models introduced in the previous examples make use of observations, caused by the hidden process, to estimate the degradation of the component. On the other hand, experts might have knowledge on how the environment influences the degradation. For example, a ball bearing tends to degrade faster when it is used in a humid environment and therefore the transition probabilities of the hidden process depend on the observed humidity. In this way, this observation (C) is rather an input for the model as shown in figure 1(d).

This type of relations can be modeled with a logical HMM (LOHMM) (Kersting et al., 2006), a inductive logic programming setting that upgrades HMMs to deal with structure. A LOHMM extends the un-

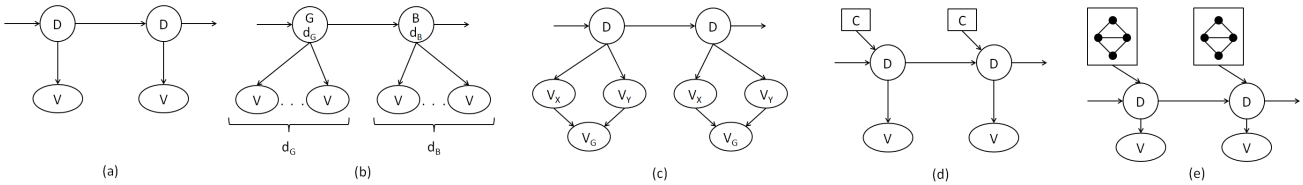


Figure 1. From left to right, we present a (a) HMM, (b) HsMM, (c) DBN, (d) LOHMM and (e) Dynamic graph for the examples defined in section 3. Note that most of the relations are probabilistic.

structured representation of transitions in an HMM by utilizing modular rules and relational information to define the transitions between states. This allows us to condition the transition probabilities on the context, for example humidity, and as such a LOHMM satisfies the fifth condition.

3.5. Probabilistic logic programming

Analogous to more complex relations between multiple observations, in some cases, more expressibility is needed to represent the structure of the context. For example, it is typically not possible to measure whether water reaches the inner side of a bearing but one can measure whether water is present at other parts of the machine. If the expert knows how the water possibly flows through the parts of the machine, he can represent this knowledge as a probabilistic graph as shown in figure 1(e). In this case the degradation of the component, and therefore the transition probabilities, is influenced by the probability of different paths in this graph.

ProbLog (De Raedt et al., 2007) is a probabilistic logic programming language which can be used to represent a probabilistic graph, compute the probability of paths in this graph and to model the hidden temporal process. Furthermore, it can be used to represent all problems introduced before and therefore satisfies our sixth requirement. Additionally, ProbLog allows us to describe knowledge in a declarative way. This is an important advantage since there is no direct mapping between a graph, represented as a set of probabilistic edges, and the nodes in a Bayesian network.

We can situate ProbLog in the more general setting of Statistical Relational Learning (SRL) (Getoor & Taskar, 2007; De Raedt & Kersting, 2003). This framework aims to integrate probabilistic reasoning mechanism with machine learning and relational or logical representations and as such allows us to incorporate different types of background or expert knowledge in a more structured and modular way. Years of research has lead to different SRL systems, each with their own strengths and limits, and therefore it is useful to compare some of these systems.

4. Experiments

As experiment, we modeled the five examples discussed in section 3, and shown in figure 1, in four state-of-the-art SRL systems: PRISM, ProbLog, Primula and Alchemy. We discuss and compare these systems and are especially interested in the trade-off between expressiveness and efficiency.

4.1. PRISM

PRISM (PRogramming In Statistical Modeling) (Sato & Kameya, 1997) is a logic-based probabilistic language implemented on top of Prolog. It combines SLD-resolution with probability calculations to compute the success probability of a query. A combination of linear tabling and restrictions on the programs makes inference and learning for a HMM as efficient as the specialized algorithms used in graphical representations.

A limiting restriction for PRISM is the exclusiveness condition which states that all disjunctive paths in a proof tree are required to be probabilistic exclusive. Because of this restriction, it is not possible to model, for example, a graph and define a path predicate. Another issue is the exponential blow-up for inference in (dynamic) Bayesian networks. In these networks inference is exponential in the number of nodes because the success probability is computed by exhaustively computing the probability of every possible world. PRISM allows the use of a junction tree such that inference is only exponential in the size of the largest clique but this requires the BN to be represented in an unnatural form and typically requires an expert.

4.2. ProbLog

The inference mechanism of ProbLog also uses SLD-resolution to find all the proofs of a query but then uses a Binary Decision Diagram (BDD), a more compact and efficient representation of the proofs. Be-

Table 1. The results of inference (I) and learning (L) for the different examples defined in section 3 and the systems introduced in section 4. HsMM1 makes use of a random (non exponential) duration and HsMM2 of a Gaussian duration.

	HMM		HsMM1		HsMM2		DBN		LOHMM		Graph	
	I	L	I	L	I	L	I	L	I	L	I	L
PRISM												
ProbLog												
Primula												
Alchemy												

cause a BDD deals with the disjoint sum problem, no restrictions need to be imposed on the program and therefore ProbLog can be used to model a graph and a path predicate. Compared with PRISM, inference in a BN is more efficient because in a BDD representation the required success probability can be computed in polynomial time.

The drawback of the expressiveness offered by ProbLog is the loss of efficiency for inference and parameter learning if it is used for temporal models like a HMM. The proofs for these models are by definition mutually exclusive and there is no need to compile them in a BDD to tackle the disjoint sum problem. In this case, compiling the proofs is unnecessary and potentially intractable for large time sequences.

4.3. Primula

Primula is an SRL system for modeling and inference with random relational structure models, more specific relational Bayesian networks (RBNs) (Jaeger, 1997). RBNs allow to define relations between different random events and combine as such a relational representation with Bayesian networks. Different from the two aforementioned systems, Primula compiles the complete network off-line based on the domains defined in a structure file and therefore allows fast inference for multiple queries. As a drawback, the domain for the network needs to be specified in the model itself which is not desired for temporal models.

4.4. Alchemy

The Alchemy system is based on the Markov logic representation (Richardson & Domingos, 2006), a combination of first-order logic and Markov networks. Markov logic can be used to represent a HMM but requires to convert probabilities into weights which is in practice very hard. Although the transition probabilities in a HMM are independent to the length of the sequence, the weights in a Markov network depend on the sequence length (Bruynooghe et al., 2009). This is undesirably because it restricts the expert to represent his knowledge of the system.

4.5. Other related systems

In the previous sections we introduced four different SRL systems but more systems exist and some of them were already compared for a (partial) different set of problems (Bruynooghe et al., 2009). Although this paper compares a larger set of systems, we deliberately chose not to test these other systems because for some of them it was already shown they cannot be used to model a HMM. Other systems are not supported anymore and are therefore also left out.

Some work specifically aims to improve the efficiency of inference and learning for temporal models with a structural or relational representation. One of them is Causal Probabilistic Time-Logic (CPT-L) (Thon et al., 2011) which defines a probability distribution over a sequence of interpretations. This technique is restricted to fully observable data and can therefore not be used to model systems where part of the data is hidden or unobserved for multiple time instances.

4.6. Comparison

The results for the four SRL systems are shown in table 1 for inference (I) and learning (L). A white box indicates the task can be executed efficiently, a gray box indicates the task can be executed but is suboptimal and a black box indicates the task cannot be executed by the system. Because of the drawbacks of Primula and Alchemy for the intended models, we do not further include them in our comparison.

Inference and parameter learning for a simple HMM is, as expected, the most efficient in PRISM. This system computes the probability distribution over the hidden states for the 50000th time instance within seconds while ProbLog fails to compute the same distribution for the 25th time instance within a minute. The efficiency of ProbLog is somewhat improved if tabling is enabled (Mantadelis & Janssens, 2010) but it still fails

to compute the distribution for the the 1000th time instance. ProbLog supports as well learning from entailment as learning from interpretations but both of them are not able to learn the parameters of the model if the observed data sequences are too long. Learning from entailment uses a k-best approach, without tabling, as underlying inference algorithm which is intractable for observation sequences larger than 25 time instances. These sequences can also not be used for the learning from interpretations setting because building the BDD is exponential and runs out of memory.

Both PRISM and ProbLog can be used to model a HsMM with a random, non-exponential, distribution over the duration (HsMM1). There is however a difference between both systems if we use them to model a HsMM with a continuous distribution over the duration (HsMM2). ProbLog supports continuous facts and accompanying predicates to model a Gaussian distribution and these are not available for PRISM. Recent work introduced a multivariate Gaussian switch for PRISM (Islam et al., 2011), but no interval checks are supported and therefore these distributions cannot be used to model a continuous duration.

In general we observe a trade-off between efficiency and expressiveness. ProbLog can be used to model the five different examples but fails to offer the same efficiency as PRISM for the temporal process. As a drawback to the efficiency offered by PRISM, it cannot be used to model a graph.

5. Conclusion

We proposed a set of requirements to model the Remaining Useful Lifetime of machines in the field of prognostics. These requirements allow us to identify the restriction enforced by probabilistic models to be able to perform efficient inference and learning. Furthermore, we experimentally compared four state-of-the-art implementations of Statistical Relational Learning approaches.

We argue that expressive, declarative models offer a language closer to the domain expert. On the other hand, efficiency is not to be ignored. The experiments have shown that SRL languages potentially allow for a trade-off within the same unifying language. Therefore, in future work, we want to develop further methods to automatically detect parts of the model where we can utilize efficient methods, thus allowing full expressiveness but with efficient inference where possible. Furthermore, more work on continuous distributions is needed because most of these distributions offered in SRL systems are still limited and experimental.

References

Bruynooghe, M., et al. (2009). An exercise with statistical relational learning systems. *International Workshop on Statistical Relational Learning*.

De Raedt, L., & Kersting, K. (2003). Probabilistic logic learning. *ACM SIGKDD Explorations Newsletter*, *5*, 31–48.

De Raedt, L., Kimmig, A., & Toivonen, H. (2007). ProbLog: A probabilistic Prolog and its application in link discovery. *In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 2468–2473).

Getoor, L., & Taskar, B. (2007). *Statistical relational learning*. MIT Press.

Islam, M. A., Ramakrishnan, C. R., & Ramakrishnan, I. V. (2011). Inference in probabilistic logic programs with continuous random variables. *CoRR*, *abs/1112.2681*.

Jaeger, M. (1997). Relational bayesian networks. *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI)* (pp. 266–273).

Kersting, K., De Raedt, L., & Raiko, T. (2006). Logical hidden markov models. *Journal of Artificial Intelligence Research*, *25*, 2006.

Mantadelis, T., & Janssens, G. (2010). Dedicated tabling for a probabilistic setting. *Technical Communications of the 26th International Conference on Logic Programming* (pp. 124–133).

Mihajlovic, V., & Petkovic, M. (2001). Dynamic Bayesian Networks: A State of the Art. *CTIT technical reports series*, *TR-CTIT-34*.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* (pp. 257–286).

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, *62*, 107–136.

Sato, T., & Kameya, Y. (1997). PRISM: A language for symbolic-statistical modeling. *In Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 1330–1335).

Thon, I., Landwehr, N., & De Raedt, L. (2011). Stochastic relational processes: Efficient inference and applications. *Machine Learning*, *82*, 239–272.

Yu, S.-Z. (2010). Hidden semi-Markov models. *Artificial Intelligence*, *174*, 215–243.