Regression Trees for System Models and Prediction

Swantje Plambeck[1], Görschwin Fey[1]

[1]Hamburg University of Technology, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany

Abstract

Today's systems are highly complex and often appear as black-box systems because of unknown internal functionalities. Thus, retrieving a model of a system is possible only based on observations of the system. We explore the usage of regression trees to learn a model of a complex system with continuous signals. Our approach for learning a regression tree uses observed inputs and outputs of a system from bounded history. We describe how to construct such a model and analyze the accuracy of predictions. Results show the applicability of regression tree models for continuous systems.

Keywords

Regression Trees, System Models, Continuous Systems

1. Introduction & Related Work

Discrete and finite models are used and necessary for many tasks like verification [1], testing [2, 3], or monitoring [4, 5]. Finding suitable models of modern systems becomes more and more challenging due to an increased complexity. We consider to learn such models to speed-up the mentioned tasks and allow modeling for black-box systems. There is a broad research interest on learning techniques in recent years where neural networks are among the prominent examples. Neural networks may handle value-continuous or value-discrete data, but they lack in finding interpretable models. Nevertheless, interpretability, especially for the assumption of a black-box system is preferable. To enable learning of discrete models, a valid discretization or abstraction of the mainly real-valued input and output signals of the system is needed [6]. Automata learning is an established approach to learn discrete models that are deterministic. There exist learning algorithms for automaton models, which learn on a given set of learning data (passive automata learning [7]) or actively ask queries to a given system to iteratively extract learning data (active automata learning [8, 9, 10]).

We consider a passive approach, but without the requirement to reset a system to an initial state. Furthermore, our approach overcomes the step of abstraction of continuous signals and learns a model directly on the real-valued samples. For this, we use regression trees – a specialization of decision trees for real-valued decision outputs. The idea of using decision trees to learn approximate models of systems was already presented in [11]. In that scope, theoretical limitations for modeling are discussed and prediction accuracies give a metric for

OVERLAY 2022: 4th Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis, November 28, 2022, Udine, Italy

swantje.plambeck@tuhh.de (S. Plambeck); goerschwin.fey@tuhh.de (G. Fey)

0000-0002-4875-5280 (S. Plambeck); 0000-0001-6433-6265 (G. Fey)

approximation. In this paper, we consider systems with real-valued input and output signals as [11], but examine capabilities of regression trees as system models.

2. Modeling Approach

2.1. Preliminaries

Definition 1 (Regression Tree). *A regression tree is a tree $T = (V, E)$ consisting of a set of nodes V and a set of edges E. The regression tree represents a function $d : \mathcal{X} \to \mathbb{R}$ that maps a feature vector $\mathbf{f} = [f_1, ..., f_m]$ from the domain $\mathcal{X}$ to a real number. The set of nodes without outgoing edges V_L are called leaf nodes or leaves. All other nodes $V \setminus V_L$ are inner nodes [12].*

We assume a binary regression tree where each inner node has exactly two outgoing edges. A regression tree T is built based on a learning set $\mathcal{L}$ consisting of a set of labeled feature vectors $(\mathbf{f}, c)$ with $\mathbf{f} \in \mathcal{X}$ and $c \in \mathbb{R}$. Every node v in the tree represents a subset $S_v \subseteq \mathcal{L}$ where each inner node splits its set into disjoint subsets S_{v_1}, S_{v_2} associated to the successors v_1, v_2 of v. The leaf nodes have a label representing the output of the regression function d. The label c_{v_l} is calculated as the mean value over all labels in the set S_{v_l} as follows

$$c_{v_l} = \frac{1}{|S_{v_l}|} \sum_{(\mathbf{f},c) \in S_{v_l}} c. \tag{1}$$

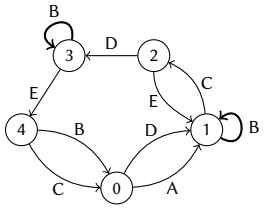
Regression trees tend to overfit the learning data. To overcome overfitting, there exist multiple strategies. Here, we apply a binning step before tree learning as part of the algorithm – meaning that equidistant discretization intervals (*bins*) are used for all real-valued features.

2.2. Learning Regression Tree Models of Systems

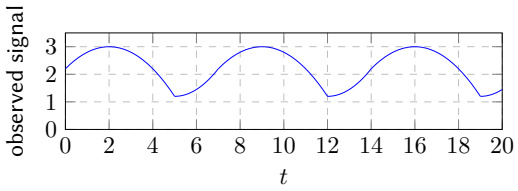
We consider a black-box system that receives a possibly multi-dimensional input $\mathbf{i}$ of dimension n and outputs a one-dimensional output $o \in \mathbb{R}$. This is a valid assumption, as usually a single value of interest is observed at a time. If several outputs are considered, multiple regression tree models can be learned in parallel. Inputs are categorical or real-valued. The observation of a system results in sequences $\tilde{\mathbf{s}} = \langle(\mathbf{i_0}, o_0), (\mathbf{i_1}, o_1), \dots\rangle$ representing sampled inputs and outputs over discrete time points. We assume to observe the system with bounded history, which is a practical requirement assuming that limited memory is available. Thus, the learning data results in observations $\mathbf{s} = \langle(\mathbf{i_0}, o_0), \dots, (\mathbf{i_N}, o_N)\rangle$ of length $N + 1$ with $N \in \mathbb{N}$. The regression tree requires a learning set $\mathcal{L}$ consisting of tuples $(\mathbf{f}, c)$. Given a sequence $\mathbf{s}$, we map to an observation for the learning set as follows

$$\mathbf{f} = \langle i_1^1, \dots, i_1^n, o_1, i_2^1, \dots, o_2, \dots, i_N^1, \dots, i_N^n\rangle, \quad c = o_N. \tag{2}$$

With this mapping, the learned regression tree represents the prediction of the next output based on the knowledge about all previous in- and outputs of the last N time steps. The regression tree's prediction accuracy is validated on an evaluation set $\mathcal{E}$ consisting of tuples of



(a) Example of a Discrete System



(b) Example of a Continuous Signal

Figure 1: Examples for Approximations of Regression Tree Models

feature vectors and a known next output $(\mathbf{f}, o_{exact})$. We use the mean prediction error (Eq. 3) as a quality metric for the learned regression tree model.

$$e_{mean} = \frac{1}{|\mathcal{E}|} \sum_{(\mathbf{f}, o_{exact}) \in \mathcal{E}} |o_{exact} - d(\mathbf{f})| \tag{3}$$

2.3. Theoretical Limitations

For an optimal model, the prediction $d_L(\mathbf{f})$ always perfectly matches the exact output o_{exact}. Regression tree models according to our approach have some theoretical limitations introducing approximations that lead to imperfect predictions. A first reason for approximations is the usage of bounded history, because the information from previous time steps might not be sufficient to fully determine the system's status and, thus, determine an upcoming output.

Example 1. *Consider that Fig. 1a shows a valid, discrete abstraction of a system, where circles 1 to 4 represent states of the system, while arrows are transitions labeled with observations from the alphabet $\{A, B, C, D, E\}$. Because of the two self-loops with observation B at state 1 and 3, bounded history does not always allow to clearly determine, in which of these states the system is and, thus, what the next observed symbol will be.*

Further limitations result from the sampling of the time-continuous signals and the fact that a regression tree has a finite set of results – thus, approximating the actual real-valued output.

Example 2. *Considering a system with an output signal as in Fig. 1b, we could achieve optimal predictions under the following conditions (1) there is no noise or measurement error, (2) the sampling of the signal is happening with a sampling rate that matches the signal's periodicity, (3) we sample the output signal always at the exact same sampling points for each observation. If any of the above conditions does not hold, approximations might be introduced to the learned model.*

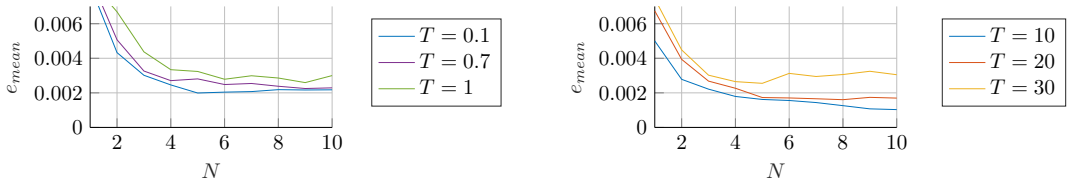
3. Experimental Evaluation

In the following, we present an empirical evaluation on example systems. For all examples 150 simulation runs are used for learning, while 50 additional simulation runs provide evaluation data. Sampling periods T are provided with respect to the simulation time unit and, thus, come

without physical unit. The number of bins for regression tree learning is set to 30 which was empirically found suitable within the interval of 10 to 100 bins.

Water Reservoir A water reservoir with one tank. The input signals are the flow rates at inflow and outflow valves while the output is the water level of the reservoir. The input and output flow rates are alternating, periodic rectangular functions describing modes of inflow and outflow. The system is modeled in Matlab and the total duration of one simulation run is 50 time units.

Boiler A temperature control system based on [13]. The input signal is the reference temperature which changes every 50 time units randomly to values from the set $\{18\,°\mathrm{C}, 19\,°\mathrm{C}, \ldots, 25\,°\mathrm{C}\}$. The output signal is the water temperature of the boiler starting at $15\,°\mathrm{C}$. The overall simulation time of one run is 1400 time units.



(a) Waterreservoir (b) Boiler

Figure 2: Relative Mean Error over Different Length N of Bounded History

Fig. 2 shows the mean error e_{mean} for the example systems with several sampling periods T over a bounded history $N = 1..10$. For comparability of the examples, the relative mean error with respect to the maximum output value of the examples is given. The water reservoir produces outputs in the range [0,5] while the boiler has an output range in [15,26]. First of all, we observe that for both examples and all sampling periods the mean error decreases over increasing N which validates the assumption that a valid next output can be predicted from past observations of the system. Furthermore, we observe a stagnation for larger N which shows that the usage of a bounded history is sufficient for the considered systems to achieve valid predictions of the future. Finally, we observe a slightly increased prediction accuracy using a smaller sampling period which is an intuitive result. Both example reach in the limit prediction errors of about 0.2% of their maximum output.

4. Conclusion & Future Work

We presented an approach to model complex systems using regression trees allowing to include continuous signals from observations of bounded history without resetting the system to an initial state. We introduced a mapping from system observations to labeled feature vectors for regression tree learning. The resulting regression tree model represents a decision rule that predicts a next output signal based on previous behavior of the system. Future work considers to evaluate further strategies for regression tree pruning and analysis of the influence of the sampling frequency as well as discretization-related approximations in modeling.

Acknowledgments

This work is partially funded by BMBF project AGenC no. 01IS22047A.

References

[1] A. Corso, R. Moss, M. Koren, R. Lee, M. Kochenderfer, A survey of algorithms for black-box safety validation of cyber-physical systems, J. Artif. Int. Res. 72 (2021). doi:10.1613/jair.1.12716.

[2] B. K. Aichernig, R. Bloem, M. Ebrahimi, M. Horn, F. Pernkopf, W. Roth, A. Rupp, M. Tappler, M. Tranninger, Learning a behavior model of hybrid systems through combining model-based testing and machine learning, in: Testing Software and Systems, 2019, pp. 3–21.

[3] S. Plambeck, J. Schyga, J. Hinckeldeyn, J. Kreutzfeldt, G. Fey, Automata learning for automated test generation of real time localization systems, 2021. arXiv:2105.11911.

[4] A. Maier, Online passive learning of timed automata for cyber-physical production systems, in: IEEE International Conference on Industrial Informatics (INDIN), 2014, pp. 60–66. doi:10.1109/INDIN.2014.6945484.

[5] F. H. Bahnsen, G. Fey, Local monitoring of embedded applications and devices using artificial neural networks, in: Euromicro Conference on Digital System Design (DSD), 2019, pp. 485–491. doi:10.1109/DSD.2019.00076.

[6] F. Howar, B. Steffen, M. Merten, Automata learning with automated alphabet abstraction refinement, in: Verification, Model Checking, and Abstract Interpretation, 2011, pp. 263–277. doi:10.1007/978-3-642-18275-4_19.

[7] K. P. Murphy, et al., Passively learning finite automata, Technical Report, Santa Fe Institute, 1995.

[8] D. Angluin, Learning regular sets from queries and counterexamples, Information and Computation 75 (1987). doi:10.1016/0890-5401(87)90052-6.

[9] M. Merten, Active automata learning for real life applications, Ph.D. thesis, Technische Universität Dortmund, 2013. doi:10.17877/DE290R-5169.

[10] H. Urbat, L. Schröder, Automata learning: An algebraic approach, in: ACM/IEEE Symposium on Logic in Computer Science, 2020, p. 900–914. doi:10.1145/3373718.3394775.

[11] S. Plambeck, L. Schammer, G. Fey, On the viability of decision trees for learning models of systems, in: Asia and South Pacific Design Automation Conference (ASP-DAC), 2022, pp. 696–701. doi:10.1109/ASP-DAC52403.2022.9712579.

[12] R. A. Berk, Classification and Regression Trees (CART), Springer, New York, 2008, pp. 1–65. doi:10.1007/978-0-387-77501-2_3.

[13] MathWorks, Bang-bang control using temporal logic, https://mathworks.com/help/simulink/slref/modeling-discrete-message-transport-systems.html, 2022. Accessed: 13.07.22.