Towards Learning From Graph Representable Formal Models

Marco Sälzer[1], Georg Siebert[2]

[1]University of Kassel, Germany
[2]University of Kassel, Germany

Abstract
We investigate different machine learning techniques to learn the semantics of graph representable formal models. Graph neural networks (GNN), which are able to learn any invariant function over graphs, became quite popular in recent years. We give empirical results on how well GNN perform in learning the semantics of nondeterministic finite automata (NFA). As our first results are positive, we discuss possible approaches to embed NFA in vector spaces such that their semantics are represented sufficiently and, thus, more classical ML techniques are applicable.

Keywords
finite automata, graph neural networks, graph kernels, vector embeddings

1. Introduction

Without any doubt, machine learning (ML), especially deep learning, has revolutionized many research fields in computer science and their applications in the past decades. Deep learning has achieved tremendous success in previously hard-to-solve tasks such as image recognition [1] or natural language understanding and processing [2]. However, this rapid rise also comes with a downside: high applicability of machine learning techniques also includes safety-critical applications and therefore a need for formal guarantees about such techniques. Unfortunately, there still is a lack of ways to combine formal methods (FM) and ML.

Recently, research on deep learning techniques for structured data gained popularity, driven by the success of *graph neural networks* (GNN), which compute functions over graphs by first learning suitable vector embeddings of the graphs [3]. Morris et al. [4] and Xu et al.[5] showed that the expressability of GNN is characterised by the combinatorial Weisfeiler-Lehman (WL) algorithm, an influential algorithm for graph isomorphism tests. It turns out that for any pair of graphs a GNN without randomly initialized parameters gives a different output if and only if (the one dimensional) WL algorithm distinguishes these graphs. However, with randomly initialized parameters GNN can learn any invariant function over graphs and, hence, are more expressive than the WL algorithm. These results and their implications are thoroughly described in [6].

OVERLAY 2021: 3rd Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis, September 22, 2021, Padova, Italy
marco.saelzer@uni-kassel.de (M. Sälzer); georg.siebert@student.uni-kassel.de (G. Siebert)
0000-0002-8012-5465 (M. Sälzer)

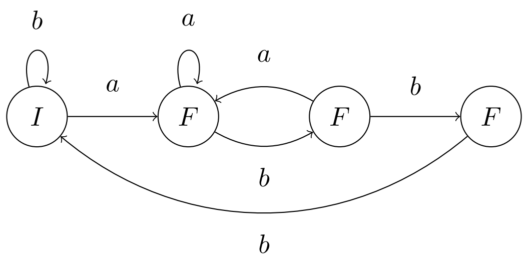


Figure 1: NFA represented as a directed and labeled graph. The language of this NFA are words w over $\{a, b\}^*$ such that w does not include the subword $abba$. The label I indicates an initial state and F a final state.

We claim that the high expressability of GNN motivates new ways to combine FM and ML in the sense that GNN are able to learn from formal models, which are representable as graphs. One of the most common of such models is the model of nondeterministic finite automata (NFA). To start our investigation, we try answering the question

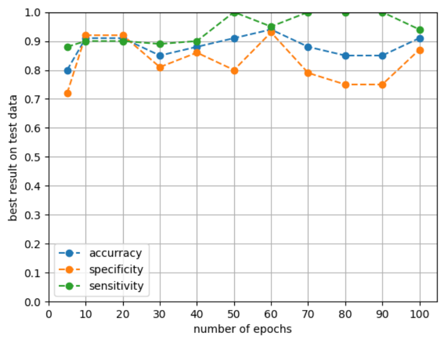
"How well do GNN perform in learning the semantics of NFA represented as graphs?".

To do this, we train GNN to distinguish NFA based on their semantics. In Sec. 2 we introduce some necessary concepts and present the results of our first experiments in Sec. 3. It turns out that GNN perform surprisingly well in distinguishing NFA, which strongly indicates that the model GNN is able to learn from graph representable formal models. This positive outcome immediately raises the question of which features the GNN learns from the graph representation of an NFA to represent its semantics in the form of real-valued vectors. As it is well-known that neural networks suffer from a lack of interpretability, we need to look at other techniques to understand which features are suitable to represent NFA semantics in vector spaces. For this, we discuss some more classical ML techniques in Sec. 4. In the last section, Sec. 5, we provide guidance on where and how this type of research can be helpful.

2. Fundamentals

Finite Automata A *nondeterministic finite automaton* (NFA) $\mathcal{A}$ for a given alphabet Σ has a finite set of states Q, for each letter in Σ a binary relation on Q and two subsets of Q called initial and final states. The language $\mathcal{A}$ is a subset of words from Σ^* that includes all words w such that there is a w-labeled path from some initial state to some final state. Following this definition, an NFA can be represented as a graph with labeled and directed edges and labeled nodes. An example is given in Fig. 1.

GNN and MPNN *Graph Neural Networks* (GNN) are a deep learning framework for the computation of functions over graphs. The general idea of GNN is that the model learns a suitable representation of the input graphs, which is then used to compute the desired function. There are several categories of GNNs like RecGNN and spectral-based or spatial-based ConvGNN



Metric	Best Overall Results
Accuracy	94%
Specificity	93%
Sensitivity	95%

Figure 2: Depiction of best experiments and corresponding accurracy, specificity and sensitivity on the test dataset per number of training epochs (line chart). The results of the overall best experiment, which happened with 60 training epochs, are depicted in the table on the right.

[3]. We focus on spatial-based ConvGNN, which follow the more general framwork of *message-passing neural network* (MPNN) [7]. GNN of this kind compute the graph representation via a message passing process between the nodes of the input graph. More details can be found in [8] or [9].

Graph Kernels *Kernels* are functions used to quantify the similarity between pairs of objects. Let O be a set of objects. A kernel is a function $O \times O \to \mathbb{R}$ which is symmetric and positive semidefinite. In the case of *graph kernels* the set O is the set of graphs. Graph kernels can be categorized into kernels based on aggregated neighbourhood information, kernels based on assignment and matching qualities, kernels based on occurring subgraph patterns and kernels based on walks and paths [10]. An important kernel of the first category is the Weisfeiler-Lehman kernel [11] which is based on the Weisfeiler-Lehman algorithm for ismorphism testing.

Graph Embedding *Graph embedding* techniques are used to find embeddings of graphs in real-valued vector spaces that preserve relevant properties of the input graph. This can be done in form of *node embeddings*, where each node is represented as a vector, *edge embeddings*, where each edge is represented as a vector, or as *whole-graph embeddings*, where the whole graph is represented as a single vector [12].

3. Experiments on Learning the Semantics of NFA With GNN

Let L be the language of all finite words over $\{a, b\}^*$ that do not contain the subword $abba$. We trained GNN for binary classification on a labeled dataset of 178 manually generated NFA. Each NFA is labeled with 1 if its language is equal to L and 0 if it is not. A corresponding NFA with label 1 is given in Fig. 1. Thus, we trained GNN to decide whether the language of an NFA, given as a graph, is equal to L or not.

In our experiments we used GNN consisting of two graph convolution layers according to [7] followed by two classic linear layers. We used a sigmoidal output activation and binary cross-entropy as loss function. In the training procedure we used *ADAM* [13] for parameter optimization and for hyper-parameter optimization we used the *Tune* [14] framework. We made a random train-validation-test split of the dataset and used 60% of the data for training, 20% for validation and 20% for testing.

Our best results are depicted in Fig. 2. In general, the GNN performed well in distinguishing NFA based on their semantics. In the overall best experiment we achieved an accuracy, specificity and sensitivity $\geq$ 90% on the test dataset. This strongly implies that the GNN model is able to learn relevant features needed to represent the semantics of NFA from their graph representations.

4. Other Approaches for Learning the Semantics of NFA

Graph kernels define similarity measures on graphs and, thus, enable the application of a wide range of machine learning techniques, so called kernel methods. Due to our first positive results with the GNN model, kernels which use neighborhood information seem promising for our application as these kernels work similarly to the techniques used in the representation learning of GNN based on the MPNN framework. In particular, it is worth taking a look at prominent variants of the WL-kernel [11] as they compute an explicit feature vector for each graph [10]. Additionally, kernels that use path information seem promising as well, as the semantics of an NFA are equivalent to the set of paths from any initial state to any final state.

Most graph kernels do not compute an explicit representation of the input graph. Therefore, direct graph embeddings [12] of NFA seem useful to create a better understanding of important features for learning the semantics of NFA. Furthermore, it allows the application of common ML techniques like Multi-Layer Perceptrons.

5. Outlook

We have begun to empirically investigate how well GNN can learn from formal models. Specifically, we have investigated how well GNN can learn the semantics of NFA from their graph representation. Based on our first experiments, it is strongly indicated that learning from formal models with GNN is possible. However, to add more robustness to these results, we intend to expand our experiments to different languages and corresponding NFA, as well as extending our dataset by size and variation.

We intend to place this research in the broader context of combining formal methods and machine learning. Robust results about ways to represent the semantics of formal models like NFA in such ways that ML techniques can be applied to them, opens up new ways to include formally specified information in typical ML applications. It remains to be seen how well such ideas generalize to different formal models.

References

[1] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Commun. ACM 60 (2017) 84–90. doi:10.1145/3065386.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, B. Kingsbury, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal Processing Magazine 29 (2012) 82–97. doi:10.1109/MSP.2012.2205597.

[3] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, P. S. Yu, A comprehensive survey on graph neural networks, IEEE Trans. Neural Networks Learn. Syst. 32 (2021) 4–24. doi:10.1109/TNNLS.2020.2978386.

[4] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, M. Grohe, Weisfeiler and leman go neural: Higher-order graph neural networks, Proceedings of the AAAI Conference on Artificial Intelligence 33 (2019) 4602–4609. URL: https://ojs.aaai.org/index.php/AAAI/article/view/4384. doi:10.1609/aaai.v33i01.33014602.

[5] K. Xu, W. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks?, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, OpenReview.net, 2019.

[6] M. Grohe, The logic of graph neural networks, in: 36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021, IEEE, 2021, pp. 1–17. doi:10.1109/LICS52264.2021.9470677.

[7] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural message passing for quantum chemistry, in: D. Precup, Y. W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, ICML 2017, volume 70 of *Proceedings of Machine Learning Research*, PMLR, 2017, pp. 1263–1272.

[8] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, OpenReview.net, 2017.

[9] W. L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 2017, pp. 1024–1034.

[10] N. M. Kriege, F. D. Johansson, C. Morris, A survey on graph kernels, Appl. Netw. Sci. 5 (2020) 6. doi:10.1007/s41109-019-0195-3.

[11] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, K. M. Borgwardt, Weisfeiler-lehman graph kernels, J. Mach. Learn. Res. 12 (2011) 2539–2561.

[12] H. Cai, V. W. Zheng, K. C. Chang, A comprehensive survey of graph embedding: Problems, techniques, and applications, IEEE Trans. Knowl. Data Eng. 30 (2018) 1616–1637. doi:10.1109/TKDE.2018.2807452.

[13] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[14] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, I. Stoica, Tune: A research platform for distributed model selection and training, arXiv:1807.05118 (2018).