Teaching the Structure of First-order Formulas to Neural Networks

Julian Parsert[1]*, Stephanie Autherith[1], and Cezary Kaliszyk[1]*

University of Innsbruck, Innsbruck, Austria
julian.parsert@gmail.com
stephanie.autherith@student.uibk.ac.at
cezary.kaliszyk@uibk.ac.at

Abstract

Logical reasoning as performed by human mathematicians involves an understanding of terms and formulas as well as transformations on them. In this paper we consider a number of syntactic and semantic properties of logical expressions. Based on these we extract and generate data sets. We develop models that encode these formulas in a continuous vector space while preserving the aforementioned properties. We train, evaluate and compare multiple models on the extracted data sets. Furthermore, we show that these models generalize to properties they have not explicitly been trained on.

Many previous examples can be found where artificial intelligence technology was applied to (interactive) theorem proving problems. While Färber [2] use simple machine learning algorithms for proof search in theorem proving, Loos et al. [6] use a deep learning approach. Also other tasks such as tactic and premise selection have been improved using different types of artificial intelligence [3, 7, 8, 4]. All of these examples and many more apply their machine learning to specific problems and extract features, engineer data etc. that precisely describes the problem at hand. We propose a learned encoding and embedding of (first-order) formulas that can later be used by more complex as well as naive models alike. Clearly, encodings of formulas need to carry syntactic and semantic information about the original formula. In addition, one would like such encodings to be relation-preserving. Ideally, the encoding of two "related" formulas will carry that relation as well. As an example, when applying these encodings to premise selection, one could imagine that useful premises would have a vector representation which are close in distance to the conjecture in question. Similarly, one could imagine application to clause selection for theorem proving, etc.

Learning Framework We propose a deep learning based encoding. Following the results from [1], we use CNNs and LSTMs based architectures. Our encoding networks are trained on char level embeddings as shown in Figure 1. This learning framework essentially consists of two main parts, the encoding network (which we are mainly interested in) and a set of classifiers. The models are trained by propagating the loss that is obtained from the classifiers back to the encoding network. Once the training phase is done, we discard the classifiers and use the encodings.

Properties The properties which are recognized in the classifiers are extracted beforehand. For now the considered properties are the subformula relation, modus ponens, term-formula distinction, well-formedness, unifiability, and alpha-equivalence. It is worth noting that there are two iterations of the subformula classification, one multilabel classification with one input

*This work is supported by the European Research Council (ERC) grant no 714034 SMART.

and one binary classification with two inputs. These formulas or pairs of formulas are fed to the learning framework where each of the formulas is first encoded by the encoding network. Then, the encodings are used as input to different types of classifiers which, as mentioned above, propagate the loss back to the encoding. The properties where chosen by considering the application of the encoding to (interactive) theorem proving. The two main focuses were 1) the structure of first-order formulas, and 2) useful properties for theorem proving. For the former we chose the properties well-formedness and subformula, whereas for the latter unifiability and modus ponens is important. The properties such as term-formula classification and alpha equivalence form an important part in both. Syntacitc and structural properties of first order logic nowadays form an important part in premise selection[5] whereas unifiability is an important property of resolution in theorem proving. We leave it up to future work to consider the minimality or the addition of these or additional properties.

Encoding Models We consider different models for our encoding network. But they can be split into a group of CNN based models and a group of LSTM based models as shown in Figure 2. All models first go through and embedding layer. After that, we have either a set of convolution/pooling layers or a set of LSTM layers depending on the model. On top of the model specific layers we put a final fully connected layer. We did not mention this explicitly yet, however we consider two types of encoding networks. Encoding networks that is functions of the form $\mathbb{N}^n \to \mathbb{R}^n$ and embedding networks, which correspond to $\mathbb{N}^n \to \mathbb{R}^m$ where $m \leq n$. The latter of which, is achieved through appending a projection layer to the encoding. Hence, we get a lower dimensional continuous representation of formulas.

Results The training data and evaluation data is split 9:1 before the training phase. The evaluation seems to confirm the results achieved in [1] where the CNNs based models outperform the LSTM based models. The best CNN based models are the ones with a fully connected layer following the convolution/pooling layers. Of the seven properties that we considered, the CNN based networks achieved anywhere between 80% and 100%. The 100% results came from the classification of terms/formulas and alpha-equivalence. Meanwhile the LSTM based models performed similarly in 4 out of the 7 considered properties. However, they perform considerably less when being tasked with classifying modus ponens, well-formedness, and subformulas. When trying to recognize a modus ponens inference step, the best LSTMs only reach an accuracy of 61%, while the best CNNs reach up to 99%. We also used the encodings and embeddings of formulas to train simpler models such as SVMs. Here, SVMs were able to recognize whether or not a term contained a variable with an accuracy of 90%. Doing a nearest neighbor analysis it also seems that the concept of variables are learned by the network.

In the future we aim for two things, adding additional properties as well as incorporating these encodings in actual theorem proving problems.

References

[1] Alexander A. Alemi, François Chollet, Niklas Eén, Geoffrey Irving, Christian Szegedy, and Josef Urban. DeepMath - deep sequence models for premise selection. In Daniel D. Lee, Masashi Sugiyama, Ulrike V. Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2235–2243, 2016.

[2] Michael Färber and Chad E. Brown. Internal guidance for satallax. In Nicola Olivetti and Ashish Tiwari, editors, *Automated Reasoning - 8th International Joint Conference, IJCAR 2016, Coimbra,*

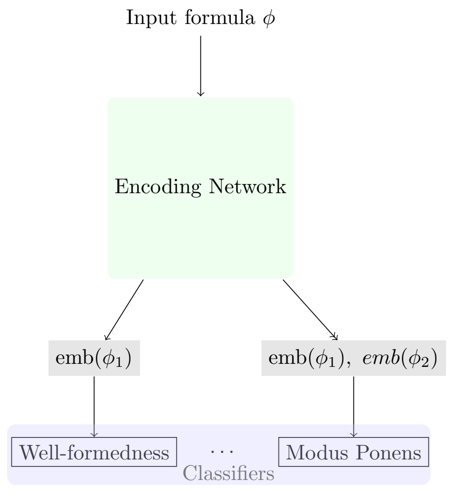

Figure 1: This graph shows the training framework we developed. The bottom area contains the classifiers that get one or more continuous representations of formulas emb(ϕ) as input. The encoding networks are described subsequently (cf. Figure 2).

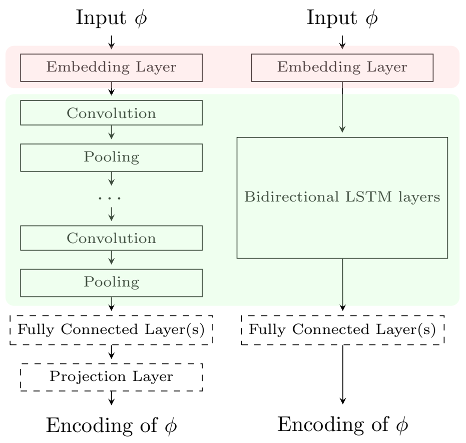

Figure 2: The encoding models we considered with the layers that the input passes through. On the left we show the CNN based models, while on the right the LSTM based models are presented. The dashed boxes describe layers that are not present in each model of that type.

Portugal, June 27 - July 2, 2016, Proceedings, volume 9706 of *Lecture Notes in Computer Science*, pages 349–361. Springer, 2016.

[3] Thibault Gauthier, Cezary Kaliszyk, and Josef Urban. Tactictoe: Learning to reason with HOL4 tactics. In Thomas Eiter and David Sands, editors, *LPAR-21, 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 46 of *EPiC Series in Computing*, pages 125–143. EasyChair, 2017.

[4] Andrzej Stanislaw Kucik and Konstantin Korovin. Premise selection with neural networks and distributed representation of features. *CoRR*, abs/1807.10268, 2018.

[5] Daniel Kühlwein, Jasmin Christian Blanchette, Cezary Kaliszyk, and Josef Urban. Mash: Machine learning for sledgehammer. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving*, pages 35–50, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[6] Sarah Loos, Geoffrey Irving, Christian Szegedy, and Cezary Kaliszyk. Deep network guided proof search. In Thomas Eiter and David Sands, editors, *LPAR-21. 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 46 of *EPiC Series in Computing*, pages 85–105. EasyChair, 2017.

[7] Yutaka Nagashima and Yilun He. PaMpeR: proof method recommendation system for Isabelle/HOL. In Marianne Huchard, Christian Kästner, and Gordon Fraser, editors, *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, ASE 2018, Montpellier, France, September 3-7, 2018*, pages 362–372. ACM, 2018.

[8] Mingzhe Wang, Yihe Tang, Jian Wang, and Jia Deng. Premise selection for theorem proving by deep graph embedding. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 2783–2793, 2017.