# Embedding Mathematical Formulas into Vector Space *

Ádám Fraknói[1], András Kornai[2], and Zsolt Zombori[3,1]

[1] Eötvös Loránd University, Budapest, Hungary
[2] Dept. of Algebra, Budapest University of Technology and Economics
[3] Alfréd Rényi Institute of Mathematics, Budapest

**Introduction** A major turning point in the history of natural language processing (NLP) was the emergence of the first semantically meaningful word embeddings into vector space [10, 2, 1, 6]. It turned out that the same representation can be extremely useful for several downstream tasks, such as translation or sentiment classification [1], that the system was not trained on explicitly. The idea behind these embeddings is really simple: a good way to characterize the meaning of a word is through the meanings of other words that tend to appear close to the given word in natural text. Arguably, finding the proper representation of words and sentences is at the core of the success that surrounds present day language models.

Numerous systems that apply Deep Learning to aid mathematical reasoning and inside the trained models they also implicly create vector embeddings for mathematical objects. There is, however, virtually no evidence that these embeddings are faithful to the semantics of the considered mathematical theory. In fact, we conjecture that the lack of proper embeddings is a major bottleneck of learning assisted theorem proving systems and overcoming this problem is one of the next major challenges. Here, we propose to make the first steps in this direction.

Graph Neural Networks (GNNs) have long been considered to be particularly suitable for representing mathematical formulas and have been used successfully in theorem provers, e.g. [7]. However, there is no analysis of the latent structure that emerges during training a GNN. Very similar to our motivation is [8], trying to create formula embeddings via learning to predict various logic specific properties, such as well-formedness or alpha-equivalence. While it is not the focus of the paper, they do look briefly at the latent space and identify some similar formulas that are embedded close to each other. [9] is a continuation that trains an autoencoder and uses the embedding to guide proof search. In contrast to these prior works, we focus more on linguistic methods taken directly from NLP. Furthermore, at this stage we care less about performance on downstream tasks and more about carefully analysing the emerging latent space. A similiar approach is followed in [4], focusing on mathematical information retrieval.

**Problem Statement** Our work focuses on applying successful embedding methods from NLP to mathematical formulas and analysing the emerging representations in terms of how well they capture the semantics of the input. We aim to identify what mechanisms from NLP work well directly and what needs to be adapted to the particularities of the formal content.

To get an idea of the differences between natural and formal languages, consider the natural and formal sentences "The weather is wonderful today" and "$3 * (6 - 2) = 24/2$". Each word in the English sentence has a rather small set of meanings and the context determines which one is applicable. The meaning of the sentence can be surprisingly well approximated by the set of the relevant meanings of the words: changing the word order only results in subtle changes. [5]

*This work has been supported by the the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory Grant no RRF-2.3.1-21-2022-00004, as well as the ELTE TKP 2021-NKTA-62 funding scheme.

(Section 1.3) provides a back-of-the-envelope estimation of the relative information content of various linguistic components in natural language, concluding that around $80-84\%$ comes from words, $12-16\%$ comes from the logical/grammatical structure and emotive content accounts for around $5-7\%$. For the arithmetic formula, however, many of the words are much more ambiguous, for example the digit 2 can mean two, twenty, two hundred etc.[1] On the other hand, formal languages have unambiguously defined compositional semantics, i.e, we know exactly how the meaning of a complex expression is built up from those of the subexpressions. In summary, the bulk of the meaning in natural language comes from words, while in formal language it comes from logical structure. It is hence an interesting – and yet open – question whether embedding methods for natural text will also work for embedding formulas.

**Current Status** Terms of a mathematical theory often have a very clear structure and one can check how well that structure manifests in the embedding space. We fix a logical theory and generate true statements as text in that theory, which are used to train a language model. We focus on the embedding of terms visualize their latent structure. For example, we can check whether terms that refer to the same entity are mapped close to each other, or if terms associated with integers align on a line. The hopeful outcome is a toolset for creating embeddings that are faithful to the semantics of the assumed theory.

As a first step, we start with variable free arithmetic formulas of the form `<exp> <rel> <exp>` where an `<exp>` is an expression built from decimal numbers and the $\{+,-,*,/\}$ operators and `rel` is a relation from $\{=,\leq,\geq\}$. An example formula looks like this:

$$((383+269)/((1*1)*(642-641)))=((571/(391/391))+81)$$

We train a BERT [3] language model on 100,000 such sentences. This model provides both static (context independent) and dynamic (context dependent) embeddings. The first encouraging result is that digits are aligned roughly on a straight line both in the static and dynamic embedding, with the exception of 0, as shown in Figure 1. Interestingly, this only holds if the training dataset only contains equalities: when ordering relations are also included, the embeddings form no recognisable pattern. As we start composing digits, however, edit distance becomes more important than semantic distance, i.e, number 110 is much closer to 120 than to 109, see Figure 2. Furthermore, the model has not learned the commutativity of addition, i.e., it does not know that $101+109$ and $109+101$ are the same, see Figure 3.

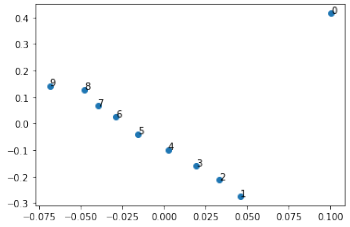

Figure 1: BERT static embedding from 0 to 9.

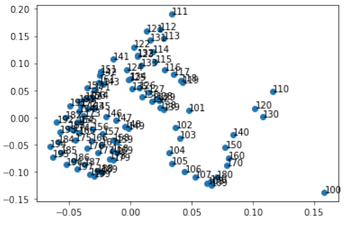

Figure 2: BERT dynamic embedding from 100 to 199.

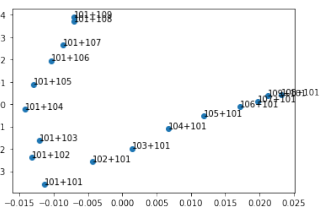

Figure 3: BERT dynamic embedding: adding numbers.

[1] Variables are the extreme example of terms whose meaning is entirely context dependent.

# References

[1] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research (JMLR)*, 2011.

[2] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Liangcai Gao, Zhuoren Jiang, Yue Yin, Ke Yuan, Zuoyu Yan, and Zhi Tang. Preliminary exploration of formula embedding for mathematical information retrieval: can mathematical formulae be embedded like a natural language? *ArXiv*, abs/1707.05154, 2017.

[5] András Kornai. *Semantics*. Springer Verlag, 2019.

[6] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[7] Miroslav Olsák, C. Kaliszyk, and Josef Urban. Property invariant embedding for automated reasoning. In *European Conference on Artificial Intelligence*, 2019.

[8] Julian Parsert, Stephanie Autherith, and Cezary Kaliszyk. Property preserving embedding of first-order logic. In Gregoire Danoy, Jun Pang, and Geoff Sutcliffe, editors, *GCAI 2020. 6th Global Conference on Artificial Intelligence (GCAI 2020)*, volume 72 of *EPiC Series in Computing*, pages 70–82. EasyChair, 2020.

[9] StanisŁaw PurgaŁ, Julian Parsert, and Cezary Kaliszyk. A study of continuous vector representations for theorem proving. *Journal of Logic and Computation*, 31(8):2057–2083, 02 2021.

[10] Hinrich Schütze. Word space. In SJ Hanson, JD Cowan, and CL Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 895–902. Morgan Kaufmann, 1993.