

Embedding Mathematical Formulas into Vector Space

Ádám Fraknói¹ András Kornai² Zsolt Zombori^{3,1}

¹Eötvös Loránd University, Budapest, Hungary

²SZTAKI Institute of Computer Science
and Dept. of Algebra, Budapest University of Technology and Economics

³Alfréd Rényi Institute of Mathematics, Budapest

8th Conference on Artificial Intelligence and Theorem Proving
September 3-8, 2023, Aussois, France



ELTE
EÖTVÖS LORÁND
TUDOMÁNYEGYETEM

Acknowledgements

This work has been supported by Hungarian National Excellence Grant 2018-1.2.1-NKP-00008, the Hungarian Artificial Intelligence National Laboratory (RRF-2.3.1-21-2022-00004) and the ELTE TKP 2021-NKTA-62 funding scheme.



Contents

1 Motivation

▶ Motivation

▶ Embeddings results

▶ Representation matters

Semantically meaningful embeddings

1 Motivation

- Entities with similar meaning should be embedded close to each other (this works well in natural text)
- Formal languages seem more difficult than natural language
 - How well can the meaning of a sentence be approximated by that of its constituent words?
E.g. "The weather is wonderful today" vs. " $3 * (6 - 2) = 24/2$ "
 - Just as in natural language, many words are ambiguous in formal language
E.g. digit 2 can mean two, twenty, two hundred etc.
- Theorem provers implicitly create vector embeddings for mathematical objects
- Not known how well such embeddings capture the theory's semantics

Previous works

1 Motivation

- Recognize the importance of a good embedding, but
- Focus on performance on downstream tasks
- Do not analyse the structure of the emerging representation (embedding)
- Do not analyze how these are learned

Example: [Parsert et al., 2020]

Our goal

1 Motivation

- Analyze the power of representations
- Understand simple tasks before moving to complicated ones
- Vision: embed logical formulas in ways that represent their semantics well
- Current work: understanding the pattern of atoms
 1. How should a good embedding for atoms of a theory should look like?
 2. What embeddings current systems can do?
 3. How can we create a good embedding?
- Using only NLP methods
 - Distally motivated by AI safety concerns [Kornai et al., 2023]
 - Considering language modeling tasks
 - Taking embeddings directly from NLP

Contents

2 Embeddings results

- ▶ Motivation
- ▶ Embeddings results
- ▶ Representation matters

Datasets

2 Embeddings results

Expression examples:

$$467 + 594 = 061$$

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

$$((3A8B3C + 2A6B9C)/((1C * 1C) * (6A4B2C - 6A4B1C))) = \\ ((5A7B1C/(3A9B1C/3A9B1C)) + 8B1C)$$

Line pattern for digits

2 Embeddings results

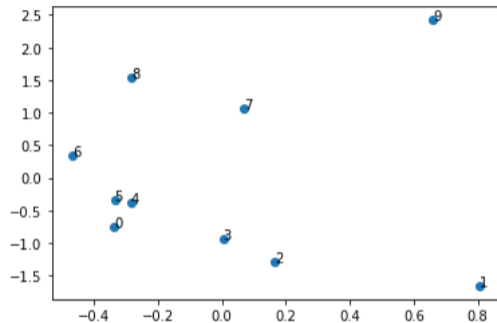


Figure: Word2vec embedding from 0 to 9

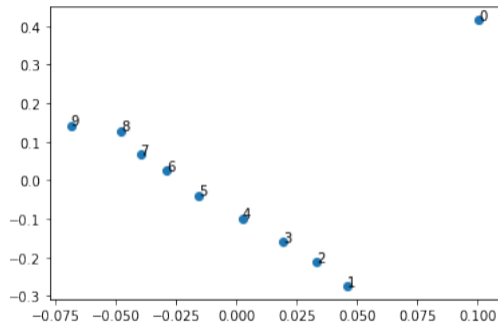


Figure: RoBERTa embedding from 0 to 9

We do not know the reason for the line pattern yet.

Further research

2 Embeddings results

- Grid pattern: numbers align in a grid as changing different digits
- Line pattern: numbers from 0-999 align in a line
- Value metric: Is $emb(A + B)$ close to $emb(C)$, where $A + B = C$
- What happens with variables?
- etc.

Contents

3 Representation matters

- ▶ Motivation
- ▶ Embeddings results
- ▶ Representation matters

How does a transformer learn to add two numbers?

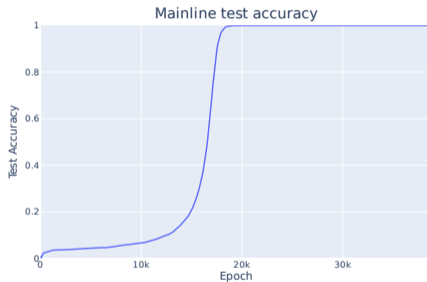
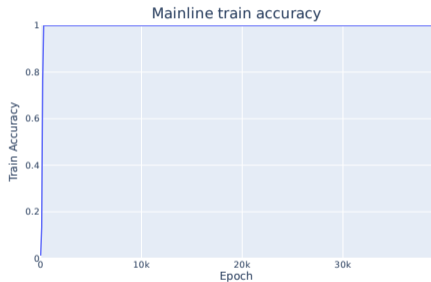
3 Representation matters

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. ICLR 2023 [Nanda et al., 2023]

- Train a 1-layer, attention only transformer to perform modulo k addition ($k=113$)
- Reverse engineer the algorithm learned by the model
- Identify key phases of the learning process:
 1. memorization (does not generalise)
 2. circuit formation (does generalise)
 3. cleanup (get rid of the stuff that does not generalise)

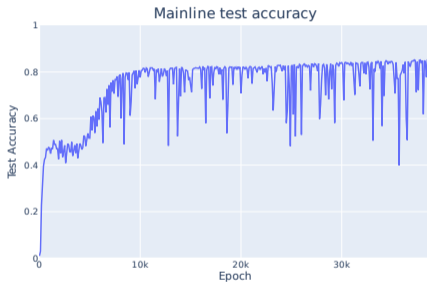
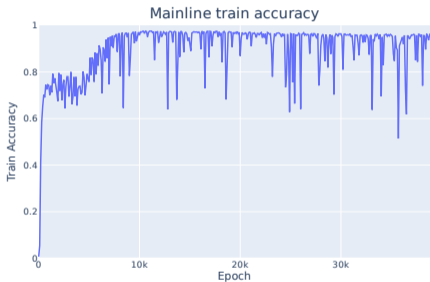
How does learning depend on the representation of the input?

How does a transformer learn to add two numbers? Onehot encoding



Memorizing training samples is quick, generalisation comes with significant delay.
This encoding is very prone to overfitting.

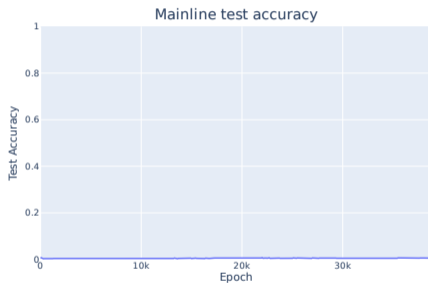
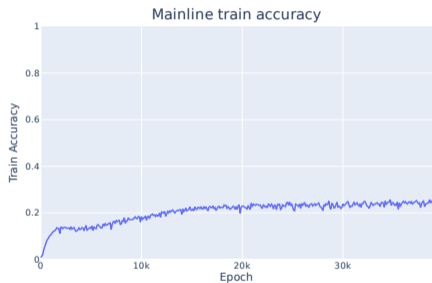
How does a transformer learn to add two numbers? Unary encoding



Memorizing training samples is a bit bumpier. No delay in generalisation. Significant generalisation gap.

This encoding supports generalisation much better.

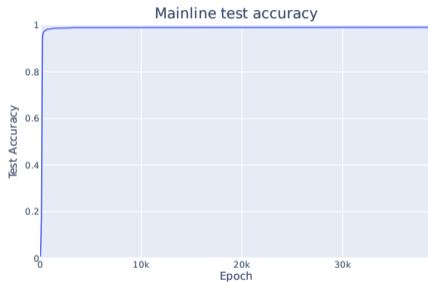
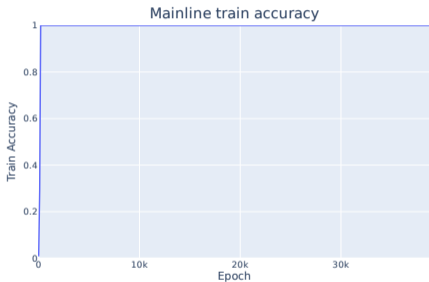
How does a transformer learn to add two numbers? Onecold encoding (1-onehot)



Learning fails, even for the training set.

This encoding makes inputs very similar in vector space. We lose sparsity.

How does a transformer learn to add two numbers? Binary encoding



Perfect, immediate learning. The input representation makes generalisation easy. This encoding presents the inputs in a spectral form. The original paper claims that such spectral form is learned by the embedding matrix.

How does a transformer learn to add two numbers? Word2vec encoding

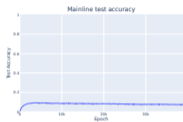
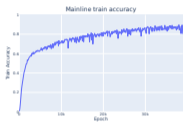
Dim Train acc

Test acc

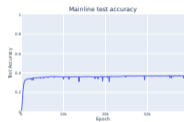
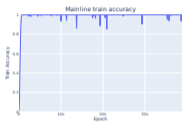
Dim Train acc

Test acc

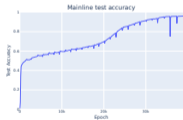
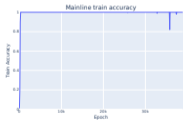
3



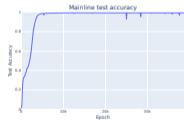
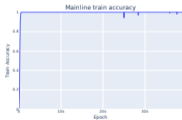
10



20






30



Performance highly depends on the embedding dimension.

For more complex downstream tasks, learned embeddings likely provide a big boost.

-  Kornai, A., Bukatin, M., and Zombori, Z. (2023).
Safety without alignment.
-  Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. (2023).
Progress measures for grokking via mechanistic interpretability.
-  Parsert, J., Autherith, S., and Kaliszyk, C. (2020).
Property preserving embedding of first-order logic.
In Danoy, G., Pang, J., and Sutcliffe, G., editors, *GCAI 2020. 6th Global Conference on Artificial Intelligence (GCAI 2020)*, volume 72 of *EPiC Series in Computing*, pages 70–82. EasyChair.

Thank you for your attention!
Q&A