Transformer as Symbolic Logic Rule-Base

Yan King Yin[1][0009−0007−8238−2442] and[2][1111−2222−3333−4444]

[1] general.intelligence@gmail.com
[2] Beijing University of Posts and Telecommunications

Abstract. This short paper shows that Transformers [4] can implement a certain flavor of symbolic-logic rules engine. As Transformers are shown to be Turing universal[3], this should come as no surprise, but such an insight could be a cornerstone for neuro-symbolic AGI.

Keywords: Transformer · Self-Attention · symbolic logic · neuro-symbolic AI

Fig. 1 shows a schematic diagram of Transformer Self-Attention, assuming the reader is familiar with its workings.

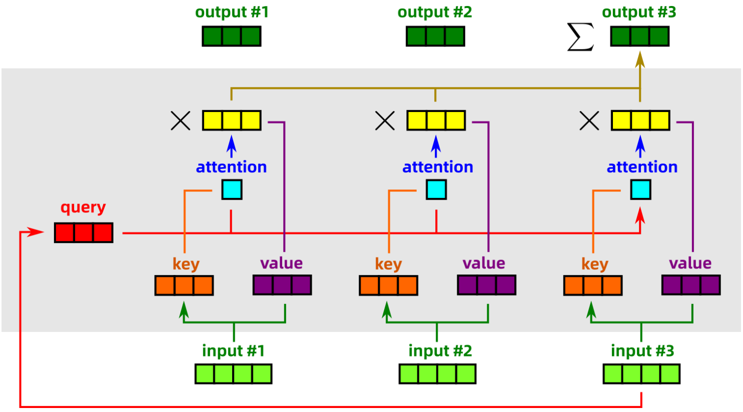

Fig. 1. Schematic of the Transformer Self-Attention

"Input" tokens are translated to Q, K, V (query, key, value)'s via matrix multiplication, which can be regarded as a kind of table look-up, or **memory store** (Fig. 2Ⓐ). From an abstract point of view, the Transformer has the structure of Fig. 2Ⓑ, which gives rise to its **equivariance** property (if input elements are swapped in a certain order, the output elements changes the same way).

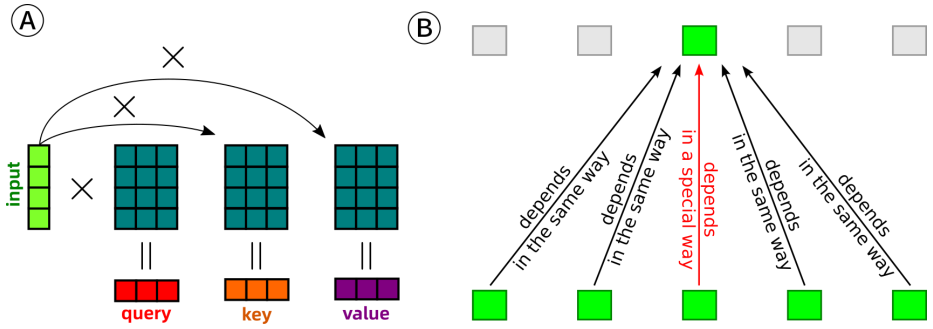

Fig. 2. Ⓐ: Q, K, V as lookup tables. Ⓑ: abstract self-attention

The equivariance property corresponds to the **exchangeability** of logic propositions:

$$A \wedge B \quad \Leftrightarrow \quad B \wedge A \tag{1}$$

For example:

$$\text{it's raining} \wedge \text{I'm heart-broken} \quad \Leftrightarrow \quad \text{I'm heart-broken} \wedge \text{it's raining} \tag{2}$$

Propositions are made up of **atomic concepts**, but at this sub-propositional level, atoms cannot be permuted freely, eg:

$$\text{I} \cdot \text{love} \cdot \text{her} \quad \neq \quad \text{she} \cdot \text{loves} \cdot \text{me} \tag{3}$$

otherwise there would be no such things as heart-breaks.

Now recall some basic notions of **classical logic-based AI**. Its basic architecture is as in Fig. 3.

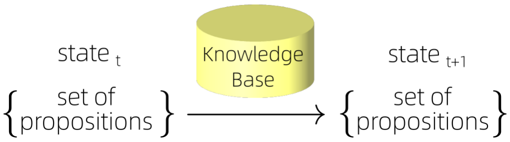

Fig. 3. Basic architecture of classical logic-based AI

There would be a huge number of rules in the Knowledge Base, and the system needs to match these rules one by one against propositions in the system's **state** (= working memory) (Fig. 4).

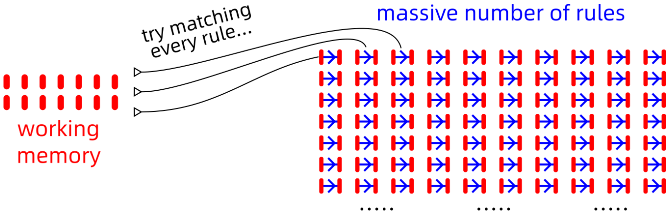

Fig. 4. Vastness of rules matching

For the Transformer, it is a kind of memory stored **between** input elements (stored as the Q, K, V matrices), and it **implicitly** plays the role of a logic rule-base (Fig. 5). This is similar to message-passing in a graph.

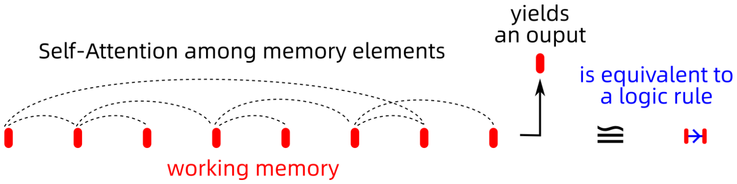
Self-Attention among memory elements
yields an ouput
is equivalent to a logic rule
≅
working memory

Fig. 5. Self-Attention as a kind of rules-matching mechanism

A crucial insight is that the **Self-Attention** mechanism had its origin in NTMs (**Neural Turing Machines**) [2]. The Turing machine needs to have a "memory tape" but in the neural setting this memory must be *differentiable*. If the memory is addressed by an index $i \in \mathbb{N}$, then it won't be differentiable. So they came up with a **content-addressable** memory mechanism where a memory matrix is looked up using the "query-key-value" method. A nice explantion of NTMs can be found in the book *Fundamentals of Deep Learning* [1].

Now consider LLMs (**Large Language Models**) such as BERT or GPT. Given a natural-language sentence, we'd like to convert or **decompose** it into a bunch of logic propositions (Fig. 6). The structure on the right of Fig. 6)Ⓐ is the

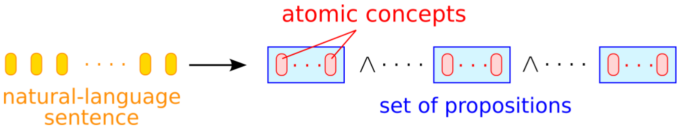

Fig. 6.

mental state of a logical AI system. It is composed of (exchangeable) propositions, which are in turn made up of atomic concepts. This 2-level structure is characteristic of all **logical** systems. Surprisingly, I found that the Transformer completely satisfies this 2-level logic structure.

On the **first layer**, a Transformer transforms each input word token into one proposition (Fig. 7Ⓐ). The crucial point here is that the propositions are composed of atoms (▭), this is achieved in the Transformer by vector **addition**, ie, **superposition** of atomic concepts. Note also that the Transformer is equivariant, so we must add “positional encoding” to each word, to indicate their ordering.

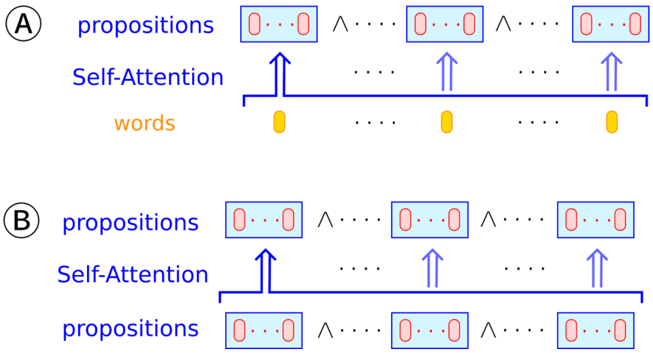

Fig. 7.

At **higher layers**, there is no need for positional encoding, and logic propositions can be freely exchanged, exactly as what happens in Transformers (Fig. 7Ⓑ).

Note that in the above, every ⇑ arrow uses the same (Q, K, V) matrices as “rule-base”, that may limit the number of rules that can be represented. To circumvent this, **Multi-Head Attention** allows to use different (Q, K, V) matrices on the same layer.

References

[1] Nikhil Buduma and Nicholas Locascio. *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. 1st. O’Reilly Media, Inc., 2017. ISBN: 1491925612.

[2] Alex Graves, Greg Wayne, and Ivo Danihelka. “Neural Turing Machines”. In: *CoRR* abs/1410.5401 (2014). arXiv: `1410.5401`. URL: `http://arxiv.org/abs/1410.5401`.

[3] Jorge Pérez, Pablo Barceló, and Javier Marinkovic. “Attention is turing complete”. In: *The Journal of Machine Learning Research* 22.1 (2021), pp. 3463–3497.

[4] Vaswani et al. “Attention is all you need”. In: (2017). `https://arxiv.org/abs/1706.03762`.