Transformer as Symbolic Logic Rule-Base

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

[1] general.intelligence@gmail.com
[2] Springer Heidelberg, Tiergartenstr. 17, 69121 Heidelberg, Germany

Abstract. This short paper shows that Transformers can implement a certain flavor of symbolic-logic rules engine. As Transformers are shown to be Turing universal[1], 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

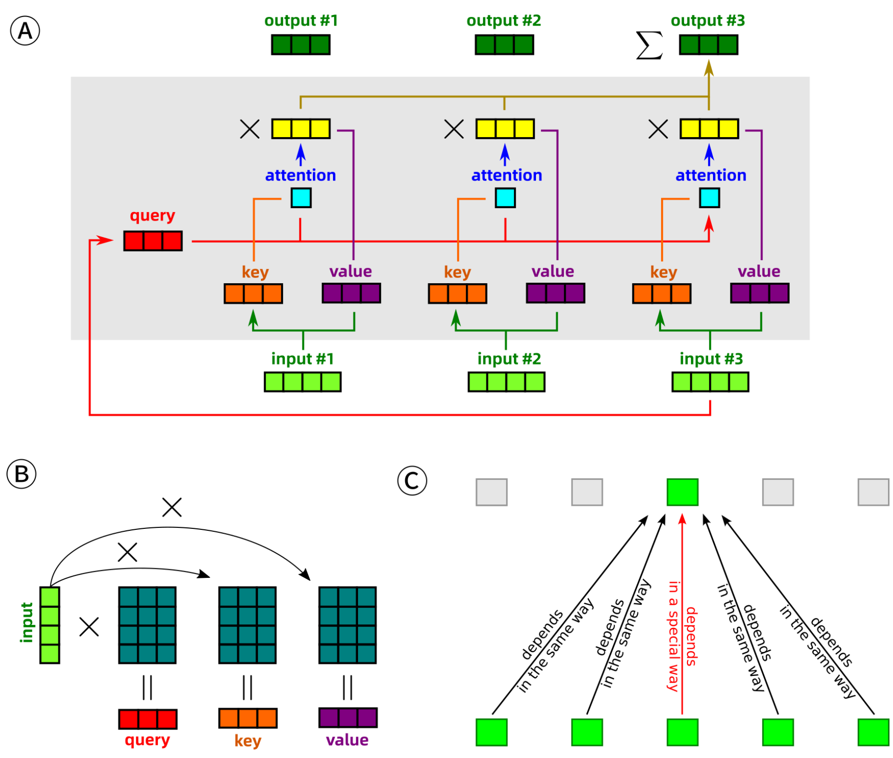

Fig. 1.

Fig 1Ⓐ shows a schematic diagram of Transformer, which serves as a refresher, assuming the reader is familiar with its workings. "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** 1Ⓑ. From an abstract point of view, the Transformer has the following structure, which gives rise to its **equivariance** property (if input elements are swapped in a certain order, the output elements changes the same way) 1Ⓒ.

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 here, at the 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.

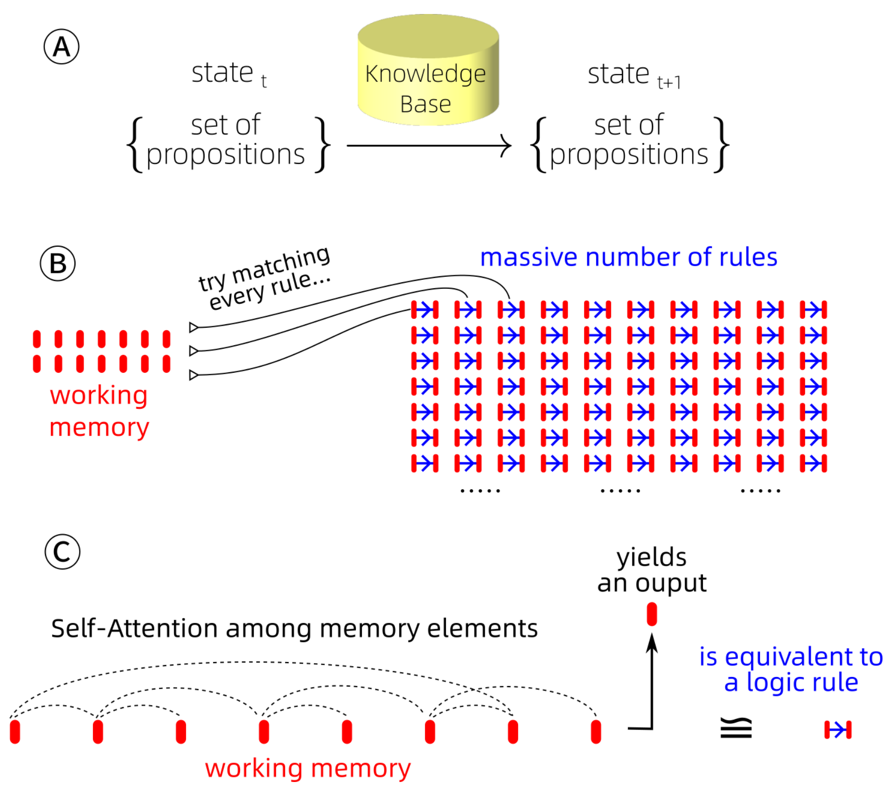

Fig. 2.

Now let's refresh a bit on **classical logic-based AI**. This is its basic architecture 2Ⓐ. 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) 2Ⓑ.

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 2Ⓒ.

A crucial insight is that the **Self-Attention** mechanism had its origin in NTMs (**Neural Turing Machines**) proposed by Graves *et al* 2014. 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* [Buduma, Locascio 2017].

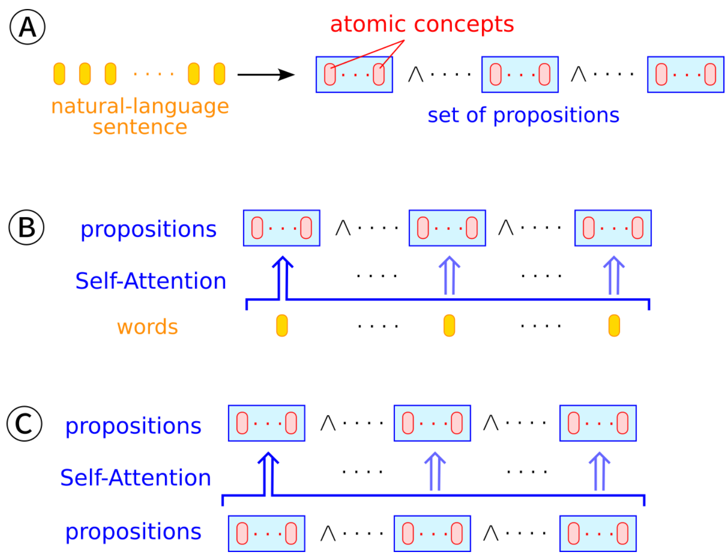

Fig. 3.

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 3. The structure on the right of (3) is a **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 3.

The crucial point here is that the propositions are composed of atoms ( ), this is achieved in the Transformer by **adding** vectors (that represent atomic concepts), ie, by **superposition**. Note also that the Transformer is equivariant, so we must add "positional encoding" to each word, to indicate their ordering.

At **higher layers**, there is no need for positional encoding, and logic propositions can be freely exchanged, exactly as what happens in Transformers 3.

Note that in the above, every $\Uparrow$ 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.

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

References

[1] Colin Wei, Yining Chen, and Tengyu Ma. "Statistically meaningful approximation: a case study on approximating turing machines with transformers". In: *arXiv preprint arXiv:2107.13163* (2021).