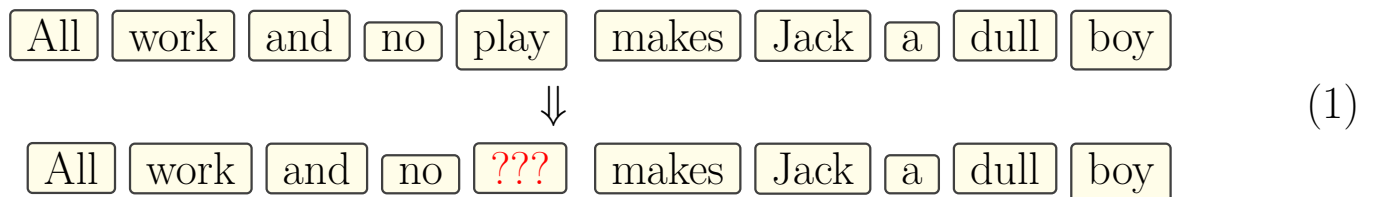# AGI for dummies

YKY

May 14, 2022

Language models and how they are trained

As of 2022, the most "intelligent" AI systems are **language models** such as BERT, GPT-3, ... and their variants.

These systems are trained to understand natural language, using a revolutionary technique by **masking out** words in a text corpus and asking the AI to **predict** them. For example:

$$\boxed{\text{All}}\ \boxed{\text{work}}\ \boxed{\text{and}}\ \boxed{\text{no}}\ \boxed{\text{play}}\ \boxed{\text{makes}}\ \boxed{\text{Jack}}\ \boxed{\text{a}}\ \boxed{\text{dull}}\ \boxed{\text{boy}}$$
$$\Downarrow \tag{1}$$
$$\boxed{\text{All}}\ \boxed{\text{work}}\ \boxed{\text{and}}\ \boxed{\text{no}}\ \boxed{\textcolor{red}{???}}\ \boxed{\text{makes}}\ \boxed{\text{Jack}}\ \boxed{\text{a}}\ \boxed{\text{dull}}\ \boxed{\text{boy}}$$

When a neural network is **forced** to make such predictions, it will gradually start to acquire knowledge similar to human's. This knowledge is stored in the (massive number of) **weights** in the neural network.

Using the masked-word trick, there is no longer the need to prepare **annotated** data to train the AI, solving one of the biggest bottlenecks in AI development.

Transformers – the most advanced AI component

The above language models are based on a key component known as **Transformers**, invented by a research team in Google:

(2)
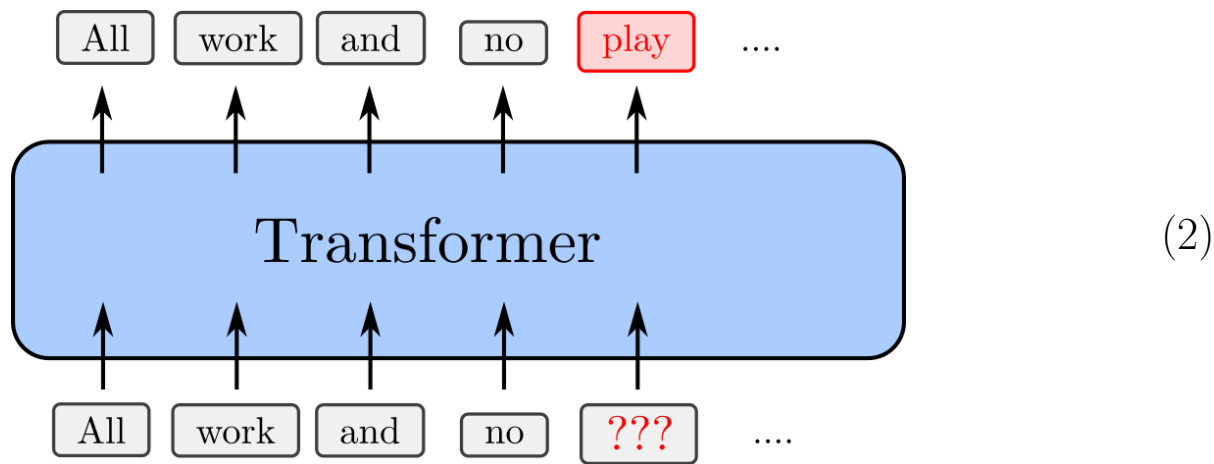
In principle, we only need a "fully-connected" neural network and train it to predict the masked words. But such a network without **additional structure** is too inefficient to learn the desired knowledge. One of the key ideas in machine learning is to give a machine some "**bias**" to make it learn faster. An example of bias is to "cut" some connections in a fully-connected neural network, so that it looks like partitioned into blocks.

The Transformer has an internal structure known as the **Attention** mechanism. It looks like this:

(3)
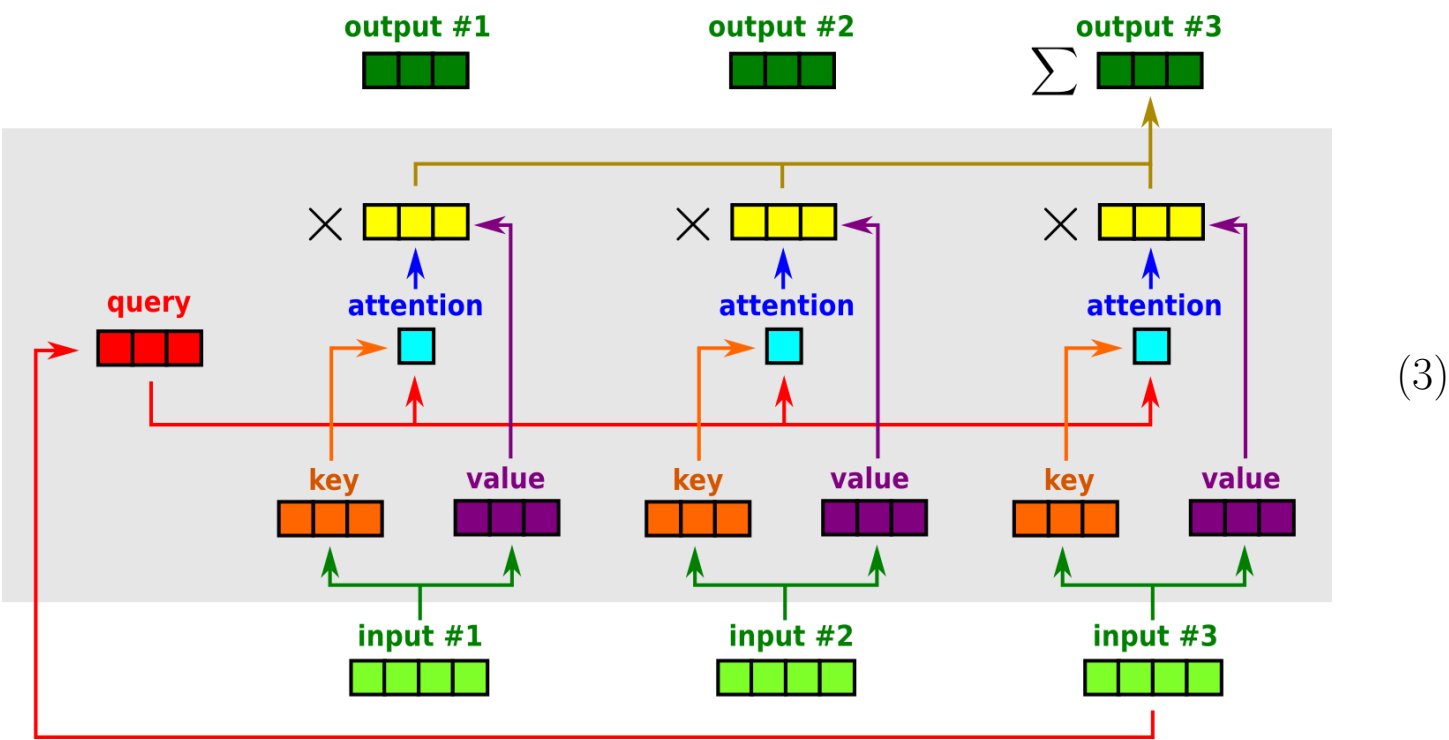
This is not widely accepted, but my hypothesis is that the Transformer performs logic reasoning.

Logic as a form of rewriting

The following are examples of **logic rules**:

"All humans are mortal":

$$\forall x.\ \text{human}(x) \Rightarrow \text{mortal}(x) \tag{4}$$

"The father's father is the grand-father":

$$\forall x, y, z.\ \text{father}(x, y) \wedge \text{father}(y, z) \Rightarrow \text{grand-father}(x, z) \tag{5}$$

The symbol $\forall$ means "for all", $\wedge$ means "and", $\Rightarrow$ means "imply".

In order for the logic formulas to work, variables such as x, y, z need to be **copied** from the left-hand-side **premise** to the right-hand-side **conclusion**. For example:

$$\text{human}(\text{Socrates}) \Rightarrow \text{mortal}(\text{Socrates})$$
$$\text{human}(\text{Plato}) \Rightarrow \text{mortal}(\text{Plato}) \tag{6}$$

These are "patterns" of a **rewriting system**. My hypothesis is that the Transformer is a kind of rewriting system. If we write the logic $\Rightarrow$ vertically as $\Uparrow$, it will look even more like the Transformer:

$$\Uparrow \quad \begin{array}{c} \text{mortal}(x) \\ \uparrow \ \uparrow \\ \text{human}(x) \end{array} \tag{7}$$

A more complicated example:

$$\Uparrow \quad \begin{array}{c} \text{grand-father}(x, z) \\ \text{father}(x, y) \wedge \text{father}(y, z) \end{array} \tag{8}$$

Notice that the bottom link $y - y$ signifies a requirement of **pattern matching**: the two y's must be substituted by the **same** object, otherwise the pattern does not match and the rule will not be applied.

# How the Transformer performs rewriting

Rete algorithm for rules-matching

In classical logic-based AI, **rules-matching** is a very inefficient problem that is solved by the **Rete algorithm**.

Naively, one's first idea is to try to match *each and every* rule to Working Memory items, which of course is very inefficient:

$$(9)$$
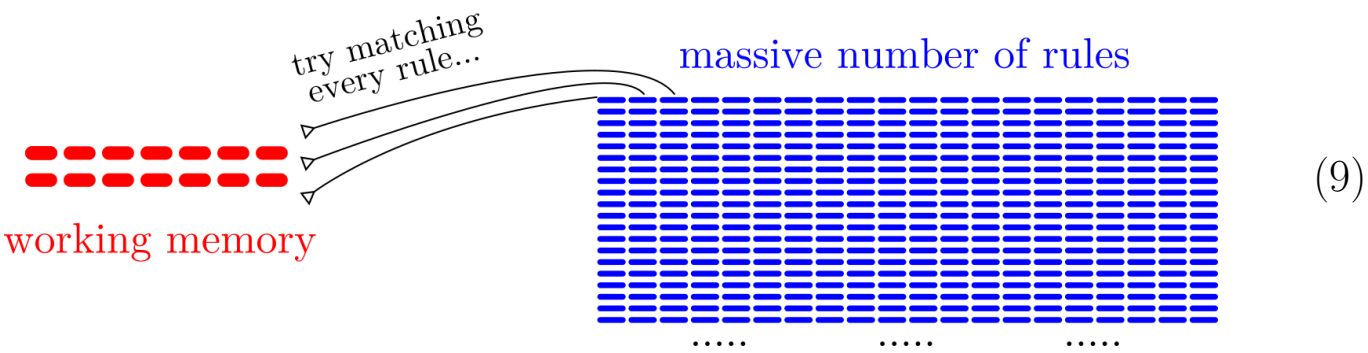
Instead, the **Rete algorithm** compiles the rules into a **classification tree**, so that each new Working Memory item is matched by "filtering" through the tree:

$$(10)$$
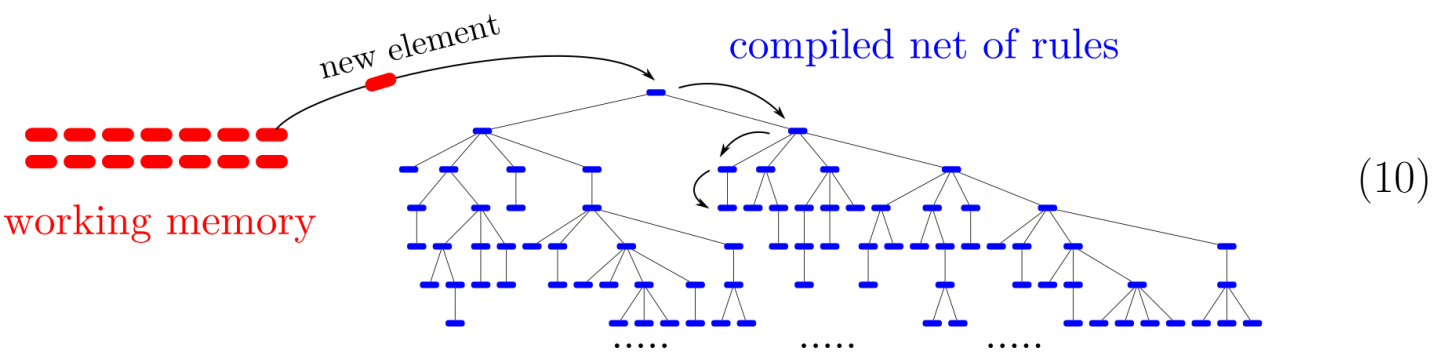
So how does the Transformer perform rules matching? It seems that rules are stored in the Q, K, V matrices. The Self-Attention mechanism is like a **message-passing** algorithm that yields a logic rule as its result:

$$(11)$$