AGI standard model
— trying to establish a consensus

YKY

March 30, 2022

0 Introduction

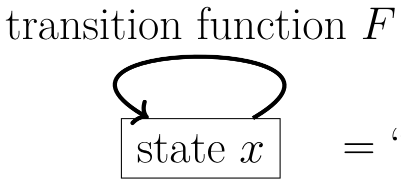
The “standard model” is a way of thinking, that may help us better understand the general theory of AGI systems.

The essence of the standard model is just to identify a **Working Memory** or “state” of the AGI system.

One benefit of our theory is that it relates Transformers / BERT / GPT to AGI systems. These language models are phenomenally intelligent, yet many people criticize them as not “truly” intelligent. The standard model suggests that they are indeed linked to AGI. There are other benefits.

1 Reinforcement learning

This is the simplest form of a **dynamical system**:



= “working memory” (1)

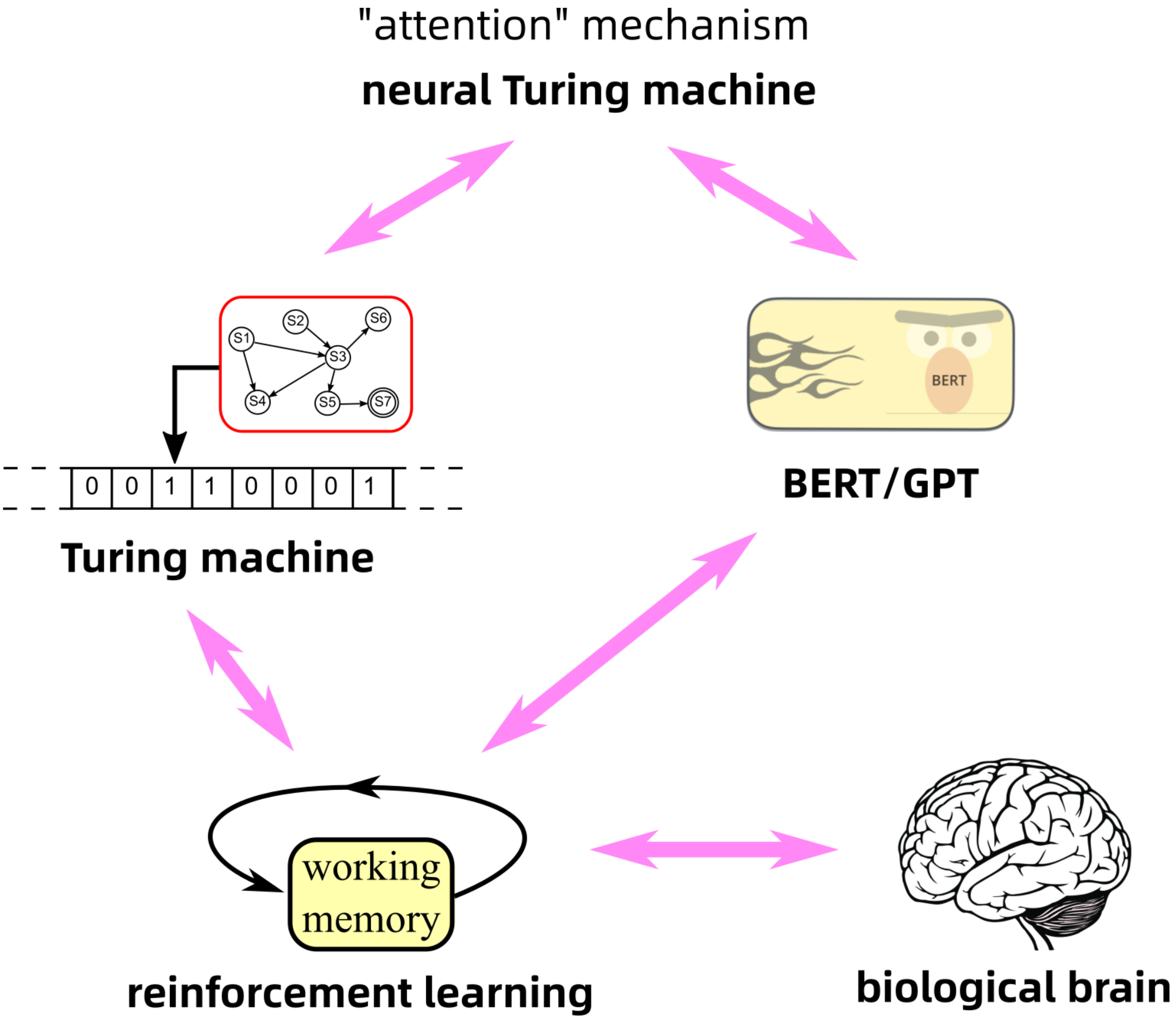
When we add an "action" or "control" variable u to it, it becomes the most basic **control system**:

$$\begin{tikzcd} \boxed{x} \arrow[loop above, "F(x,u)"] \end{tikzcd} \tag{2}$$

which is the setting for Dynamic Programming or **Reinforcement Learning**. The optimal solution for such systems is governed by the **Hamilton-Jacobi-Bellman equation**.

I call this the "standard model" because of the extreme simplicity of this setup, and that I don't know of other alternative models that deviate much from it.

Next we look at how the standard model relates to other important aspects of AGI:

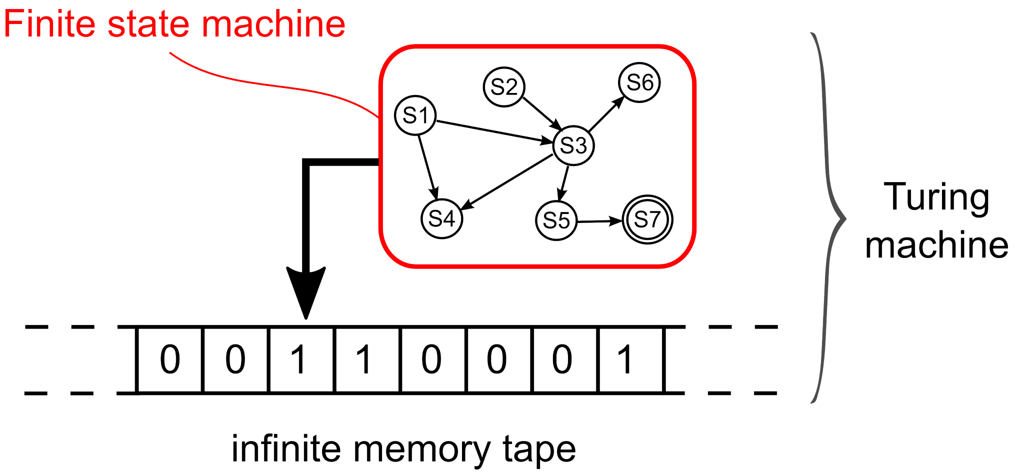


(3)

2 Neural Turing Machine and BERT

The **attention mechanism** was first proposed in the "**Neural Turing Machine**" paper by Graves *et al.*

Recall that a Turing machine is a **Finite State Machine** augmented with a **Memory Tape**:

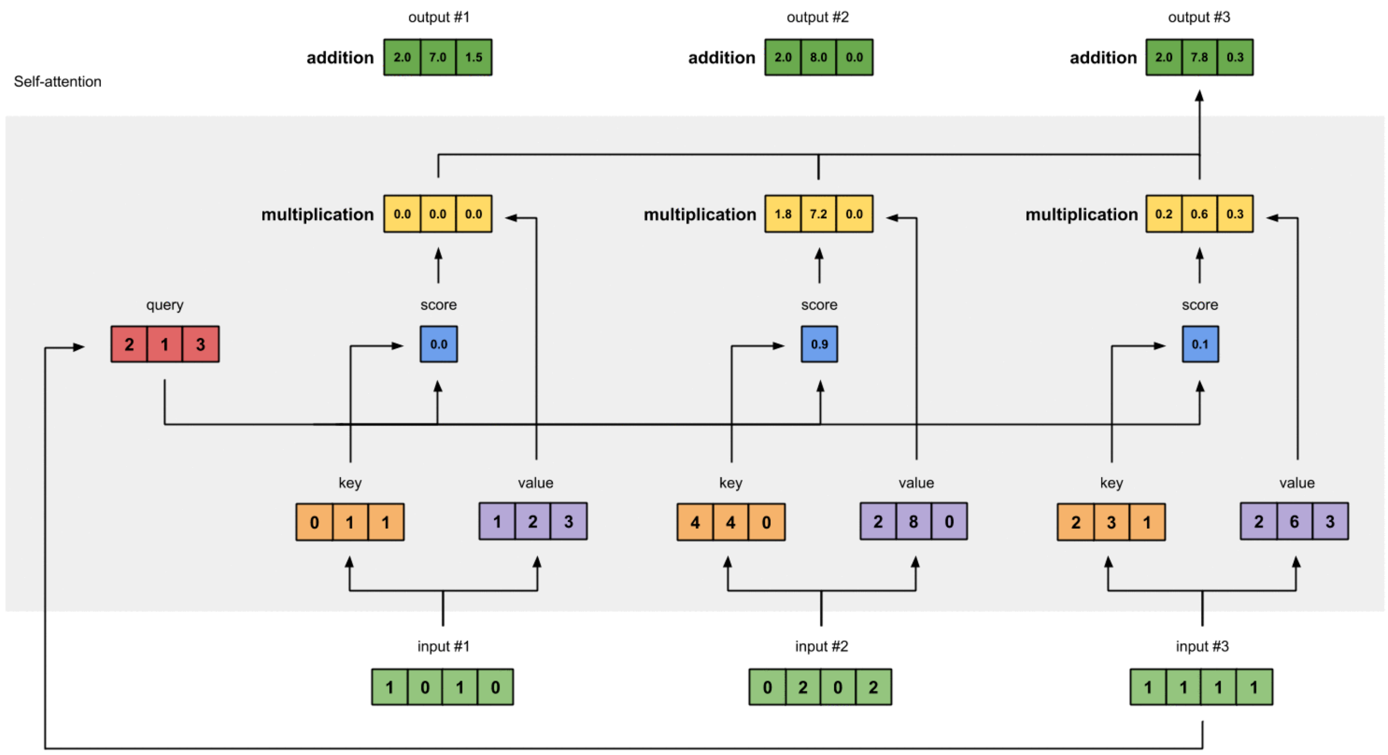


(4)

In Neural Turing Machines, Graves *et al* proposed the attention mechanism for an RNN to read and write from a **Memory Matrix**, using a content-based addressing method.

The research done by Olah *et al*, in their paper *A Mathematical Framework for Transformer Circuits*, is very helpful towards understanding Transformers and Self-Attention. The paper is technically quite challenging, but thanks to the guidance of professor Xiao Da from Beijing I was able to understand the main ideas. Here I offer some pointers to help others understand the paper, without explaining it in full details.

First, it is helpful to recall this diagram describing Self-Attention (ripped from a blog article on the web):



(5)

The Self Attention A is held constant.

Just as a **linear** map is defined between two vector spaces:

$$F : U \to V \tag{6}$$

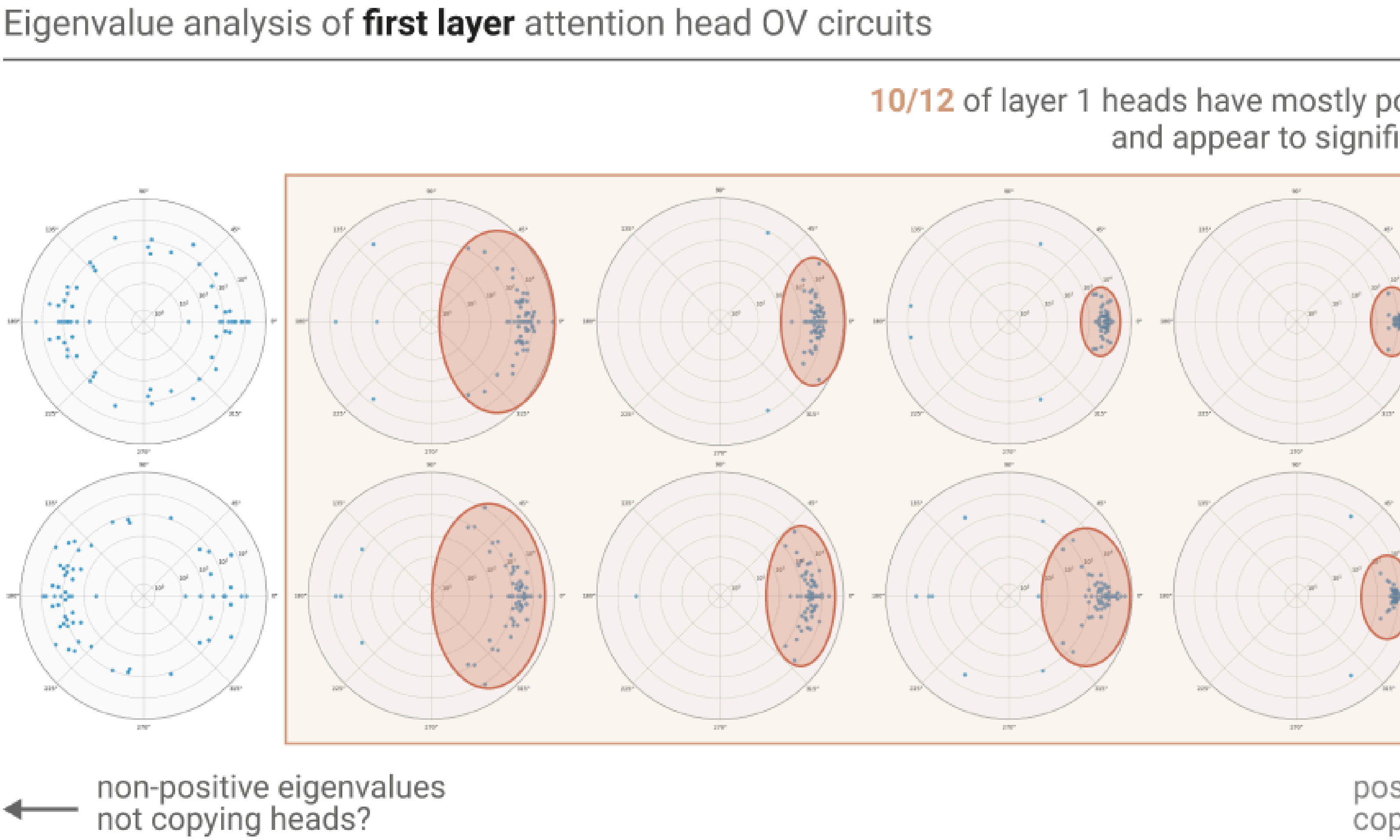
one can define a **bi-linear** map:

$$\Phi : V \times W \to U \tag{7}$$

that is linear in both its first and second arguments. The tensor product $\otimes$ is the **universal** bi-linear map.

The tensor product generalizes to the mathematically important **monoidal categories**.

Eigen-value plots show that "copy" maps are ubiquitous in Transformers. These may be interpreted as **universally-quantified** (∀) logic rules.



Eigenvalue analysis of **first layer** attention head OV circuits

10/12 of layer 1 heads have mostly po
and appear to signifi

non-positive eigenvalues
not copying heads?

pos
cop

(8)

3 Relation to the biological brain

There are two distinct aspects in the brain:

- **Short-term** or Working Memory is the **electric activation** of neuronal populations.
- **Long-term** memory is stored as **synaptic strengths**, established by synaptic formation and strengthening. The transfer from STM to LTM is called **memory consolidation**.

One theory has it that the prefrontal cortex maintains a number of "thoughts" with sub-populations or, perhaps, with **micro-columns**. These activated sub-populations are in competition with each other, through **lateral inhibition**. The thought(s) that win are the thoughts we retain – they "make sense".

4 Abductive reasoning

Abductive reasoning is basically just **bidirectional** inference.

When a system has both forward and backward connections, it forms a loop and its dynamics is likely to produce "resonance". This harks back to the ART (Adaptive Resonance Theory) proposed by Grossberg and Carpenter beginning in the 1980s.

Such resonance behavior can be viewed as the system seeking to minimize an energy, ie, trying to find the "best explanation" to a set of facts.

5 Dealing with assumptions

Example of an assumption: "If I play move x now, I will checkmate in 3 moves".

Suppose M, N are proofs of $M : \phi \to \psi$ and $N : \phi$. Then the proof of ψ would be the application of M to N, denoted as $@(M, N)$ or simply MN.

The assumption rule (Ax):

$$\Gamma, \phi \vdash \phi \qquad \text{(Ax)} \tag{9}$$

for example can be written as:

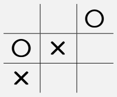
$$x : \phi,\ y : \psi \vdash x : \phi \qquad \text{(Ax)} \tag{10}$$

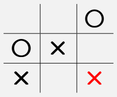
which is why we say that an assumption is a **λ-variable**.

Discharging an assumption (ie, using the →I rule) results in a λ-term.

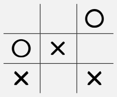
An AGI needs the ability to place an implication $\phi \to \psi$ into working memory, and to prove it using the $(\to \text{I})$ rule, ie, by making an assumption.

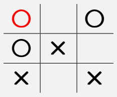
Tic Tac Toe example

Assume the current board is  and it's ✕'s turn to play.

✕ can do the "double fork" by playing .

But how can an AGI know (or prove) this?

If the current board is  then a double fork exists. We need a predicate to detect double forks.

We need to reason that even if ○ plays the "blocking" move  , ✕ can still win.

We can easily express the conditions for ✕-can-win, but the difficult part is to make the assumption in red, in other words:

$$\text{red-move} \rightarrow \times\text{-can-win} \tag{11}$$

The difficulty lies in that the LHS is **not true** under the current facts. This conditional statement must be proven by, first, assuming the LHS, and then deriving the RHS. Then the assumption is **discharged** and the conditional statement is proven, via the ($\rightarrow$ I) rule.

In the old days, in classical AI, assumptions are handled with **Truth Maintenance Systems** that keep track of inference traces symbolically. These systems can get quite complicated with the need to track multiple assumptions. For example, when we plan a bank robbery, we need to consider many possible forking scenarios.

Algorithm: Put the assumption A in Working Memory. Make inferences, marking all conclusions with $A \rightarrow *$. When enough conclusions are obtained, remove the assumption A from Working Memory.