AGI via Combining Logic with Deep Learning

甄景贤 (King-Yin Yan)

General.Intelligence@Gmail.com

Abstract. An integration of deep learning and logic is proposed, based on the Curry-Howard isomorphism. Under this interpretation, it turns out that Google's BERT, which many currently state-of-the-art language models are derived from, can be regarded as a special form of logic. Moreover, this structure can be incorporated under a reinforcement-learning framework to form a minimal AGI architecture. We also mention some insights gleaned from category and topos theory that may be helpful to practitioners of AGI.

Keywords: deep learning, symbolic logic, logic-based AI, neural-symbolic integration, Curry-Howard isomorphism, category theory, topos theory, fuzzy logic

(To the editor: Diagrams in this paper can be re-organized in the traditional format, please contact the author if this is desired. Some references are yet to be filled in.)

0 Introduction

The results in the present paper does not make use of category theory in any significant way (nor the Curry-Howard isomorphism, for that matter). Its main accomplishment is to express AGI in the categorical language. To the layperson the concepts of category theory (such as pullbacks, adjunctions, toposes, sheaves, ...) may be difficult to grasp, but they are the mathematician's "daily bread". We hope that describing AGI in categorical terms will entice more mathematicians to work on this important problem.

Secondly, an abstract formulation allows us to see clearly what is meant by "the mathematical structure of logic", without which logic is just a collection of strange rules and axioms, leaving us with a feeling that something may be "amiss" in our theory.

0.1 The Curry-Howard Isomorphism

As the risk of sounding too elementary, we would go over some basic background knowledge, that may help those readers who are unfamiliar with this area of mathematics.

The Curry-Howard isomorphism expresses a connection between logic **syntax** and its underlying **proof** mechanism. Consider the mathematical declaration of a **function** f with its domain and co-domain:

$$f : A \to B. \tag{1}$$

This notation comes from type theory, where A and B are **types** (which we can think of as sets or general spaces) and the function f is an **element** in the function space $A \to B$, which is also a type.

What the Curry-Howard isomorphism says is that we can regard $A \to B$ as a **logic** formula $A \Rightarrow B$ (an implication), and the function f as a **proof** process that maps a proof of A to a proof of B.

The following may give a clearer picture:

$$\begin{array}{ll} \boxed{\text{logic}} & A \Longrightarrow B \\ & \text{- - - - - - - - - -} \\ \boxed{\text{program}} & \blacksquare \overset{f}{\longmapsto} \blacksquare \quad . \end{array} \tag{2}$$

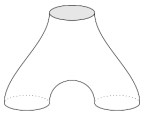
What we see here is a logic formula "on the surface", with an underlying proof mechanism which is a **function**. Here the ■'s represent proof objects or witnesses. The logic propositions A and B coincide with the **domains** (or **types**) specified by type theory. Hence the great educator Philip Wadler calls it "propositions as types". [1] Other textbooks on the Curry-Howard isomorphism include: [5] [**Simmons2000**] [**Thompson1991**].

[1] See his introductory video: https://www.youtube.com/watch?v=IOiZatlZtGU .

The gist of our theory is that Deep Learning provides us with neural networks (ie. non-linear functions) that serve as the proof mechanism of logic via the Curry-Howard isomorphism. With this interpretation, we can impose the mathematical structure of logic (such as symmetries) onto neural networks. Such constraints serve as **inductive bias** that can accelerate learning, according to the celebrated "No Free Lunch" theory [**Wolpert1997**].

In particular, logic propositions in a conjunction (such as $A \wedge B$) are commutative, ie. invariant under permutations, which is a "symmetry" of logic. This symmetry essentially decomposes a logic "state" into a set of propositions, and seems to be a fundamental feature of most logics known to humans. Imposing this symmetry on neural networks gives rise to symmetric neural networks, which can be easily implemented thanks to separate research on the latter topic. This is discussed in §3.

As an aside, the Curry-Howard isomorphism also establishes connections to diverse disciplines. Whenever there is a space of elements and some operations over them, there is a chance that it has an underlying "logic" to it. For example, in quantum mechanics, Hilbert space and Hermitian operators. Another example: in String Theory, strings and cobordisms between them (The following is the famous "pair of pants" cobordism, representing a process in time that splits one string into two):



(3)

1 Prior Research

1.1 Neuro-Symbolic Integration

There has been a long history of attempts to integrate symbolic logic with neural processing, with pioneers such as Ron Sun, Dov Gabbay, among others. We consider two approaches below.

Pascal Hitzler (*et. al.*)'s "Core Method" [1] is model-based (as in model theory in logic). An interpretation $\mathcal{I}$ is a function that assigns truth values to the set of all possible ground atoms. To a logic program P is associated a semantic operator $\mathcal{T}_P : \mathcal{I}_P \to \mathcal{I}_P$, where $\mathcal{I}_P$ is an interpretation endowed with the topology of the Cantor set. Then P is approximated by a neural network $f : \mathcal{I}_P \to \mathbb{R}$.

∂-ILP [**Evans2017**] is also model-based, and moreover it is focused on the learning problem. A valuation is a vector $[0, 1]^n$ mapping each ground atom to a real number $\in [0, 1]$. Each clause is attached with a Boolean flag (actually relaxed to be a continuous value) to indicate whether it is included in the results or not. From each clause c one can generate a function $\mathcal{F}_c$ on valuations that implements a single step of forward inference. Then gradient descent is used to learn which clauses should be included.

The author believes that representing and learning relational knowledge is perhaps the final piece of the puzzle in AGI research. We would like to mention Geoffrey Hinton's recent **GLOM theory** [**Hinton**], which addresses the problem of representing a hierarchy of visual structures.

1.2 Cognitive Architectures and Reinforcement Learning

Reinforcement Learning (RL). In the 1980's, Richard Sutton [6] introduced reinforcement learning as an AI paradigm, drawing inspiration from Control Theory and Dynamic Programming. In retrospect, RL already has sufficient generality to be considered an AGI theory, or at least as a top-level framework for describing AGI architectures.

Relation to AIXI. AIXI is an abstract AGI model introduced by Marcus Hutter in 2000 [**Hutter2005**]. AIXI's environmental setting is the external "world" as observed by some sensors. The agent's internal model is a universal Turing machine (UTM), and the optimal action is chosen by maximizing potential rewards over all programs of the UTM. In our (minimal) model, the UTM is constrained to be a neural network, where the NN's **state** is analogous to the UTM's **tape**, and the optimal weights (program) are found via Bellman optimality.

Relation to Quantum mechanics and Path Integrals. At the core of RL is the Bellman equation, which governs the update of the utility function to reach its optimal value. This equation (in discrete time) is equivalent to the Hamilton-Jacobi equation in differential form. Nowadays they are unified as the Hamilton-Jacobi-Bellman equation,

under the name "optimal control theory" [3]. In turn, the Hamilton-Jacobi equation is closely related to the Schrödinger equation in quantum mechanics:

$$\boxed{\text{Bellman eqn.}} \; - - - \; \boxed{\text{Hamilton-Jacobi eqn.}} \; - - - \; \boxed{\text{Schrödinger eqn.}} \tag{4}$$

but the second link is merely "heuristic"; it is the well-studied "quantization" process whose meaning remains mysterious to this day. Nevertheless, the path integral method introduced by Richard Feynmann can be applied to RL algorithms, eg. [**Kappen2007**].

The Hamilton-Jacobi equation gives the RL setting a "symplectic" structure []; Such problems are best solved by so-called symplectic integrators. Surprisingly, in the RL / AI literature, which has witnessed tremendous growth in recent years, there is scarcely any mention of the Hamilton-Jacobi connection, while the most efficient heuristics (such as policy gradient, Actor-Critic, etc.) seem to exploit other structural characteristics of "the world".

2 The Mathematical Structure of Logic

Currently, the most mathematically advanced and satisfactory description of logic seems to base on category theory, known as categorial logic and topos theory. This direction was pioneered by William Lawvere in the 1950-60's. The body of work in this field is quite vast, but we shall briefly mention some points that are relevant to AGI. A more detailed tutorial on categorical logic, with a focus on AGI, is in preparation [**Yan2021**].

2.1 Predicates and Dependent Type Theory

The Curry-Howard isomorphism identifies *propositional* intuitionistic logic with type theory. As such, the arrow $\to$ in type theory is "used up" (it corresponds to the implication arrow $\Rightarrow$ in intuitionistic logic). However, predicates are also a kind of functions (arrows), so how could we accomodate predicates in type theory such that Curry-Howard continues to hold? This is the idea behind Martin Löf's dependent type theory.

The expressiveness of predicate logic (in one form or another) is a highly desirable feature for AGI knowledge representation. So it seems necessary to incoporate dependent type theory into our logic. From a categorical perspective, predicates can be regarded as **fibers** over a base set; This is the treatment given in Bart Jacob's book [2]. Thus category theory gives us more insight into the (predicate) structure of logic, though it is as yet unclear how to make use of this particular idea.

2.2 (Fuzzy?) Topos Theory

The author's previous paper [7] proposed a fuzzy-probabilistic logic where probabilities are distributed over fuzzy truth values. To this day, he still believes that regarding fuzziness as a generalization of binary truth is philosophically sound. Thus it behoves to develop a generalization of standard topos theory to the fuzzy case.

The most important commutative diagram in Topos theory is this one:

$$\begin{array}{ccc} X & \xrightarrow{!} & 1 \\ {\scriptstyle m}\big\downarrow & & \big\downarrow{\scriptstyle \text{true}} \\ Y & \xrightarrow[\chi_m]{} & \Omega \end{array} \tag{5}$$

It can be understood as saying that every set is a pullback of the true map, in analogy to the idea of a "moduli space". Following this idea, is it true that every fuzzy set should be the pullback of the fuzzy true map? It seems that the theory of fuzzy topos is still an open research problem [**Jardine2019**] [**Vickers2010**].

3 Permutation Symmetry and Symmetric Neural Networks

From the categorical perspective, we make the following correspondence with logic and type theory:

$$\begin{array}{lcccl} \boxed{\text{product}} & A \times B & \leftrightsquigarrow & A \wedge B & \boxed{\text{conjunction}} \\ \boxed{\text{function}} & A \to B & \leftrightsquigarrow & A \Rightarrow B & \boxed{\text{implication}}. \end{array} \tag{6}$$

One basic characteristic of (classical) logic is that the conjuction $\wedge$ is **commutative**:

$$\mathsf{P} \wedge \mathsf{Q} \quad \Leftrightarrow \quad \mathsf{Q} \wedge \mathsf{P}. \tag{7}$$

This remains true of probabilistic logic, where $\wedge$ and $\vee$ are unified as conditional probability tables (CPTs) for the nodes in Bayesian networks.

Once we know the symmetry, the question is how to impose this symmetry on deep neural networks. Interestingly, the answer already comes from an independent line of research (namely, PointNet [4] and Deep Sets [8]) that deals with visual object recognition of point clouds, eg:

 (8)

In a point cloud, it does not matter the order in which the points are presented, as inputs to the classifier function. Such a function needs to be permutation invariant to a huge number of points.

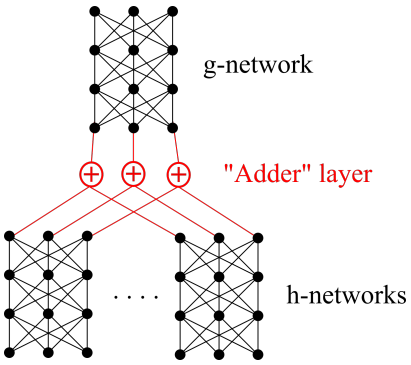
From [8]: the **Kolmogorov-Arnold representation theorem** states that every multivariate continuous function can be represented as a sum of continuous functions of one variable:

$$f(x_1, ..., x_n) = \sum_{q=0}^{2n} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right) \tag{9}$$

It can be specialized to such that every symmetric multivariate function can be represented as a sum of (the same) functions of one variable:

$$f(x_1, ..., x_n) = g(h(x_1) + ... + h(x_n)) \tag{10}$$

This leads to the following implementation using neural networks:

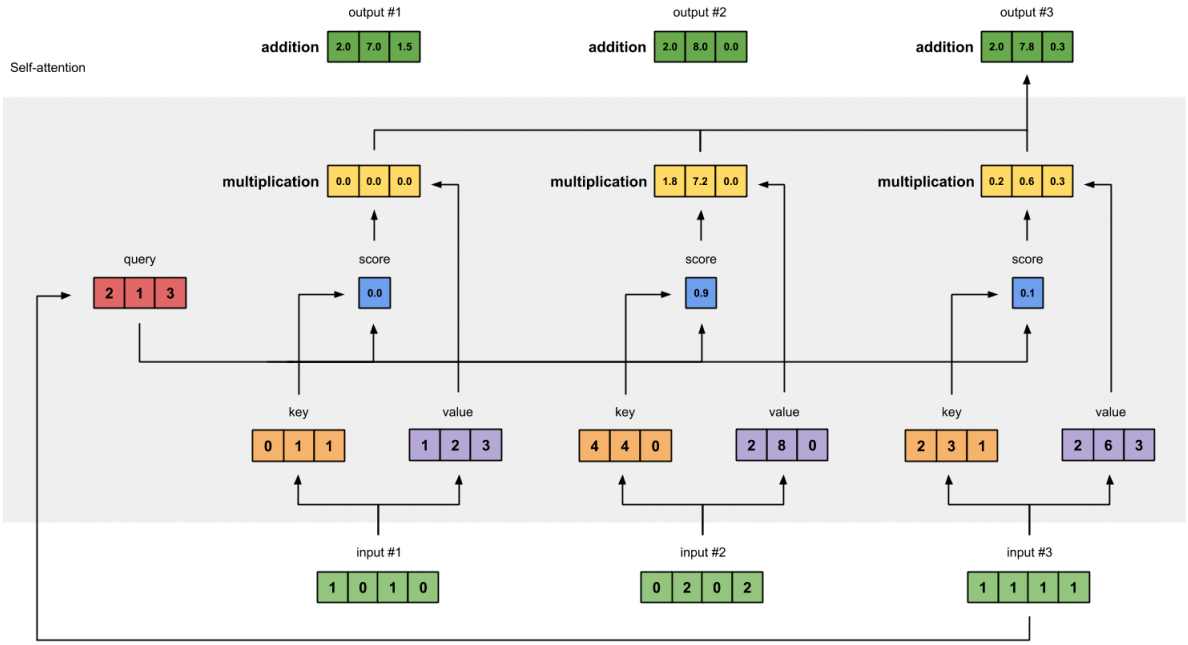


(11)

And this can be easily implemented with a few lines of Tensorflow, see §5.

3.1 Why BERT is a Logic

In the following diagram [2], observe that the Transformer is permutation-invariant (or more precisely, **equivariant**). That is to say, for example, if input #1 and #2 are swapped, then output #1 and #2 would also be swapped:



(12)

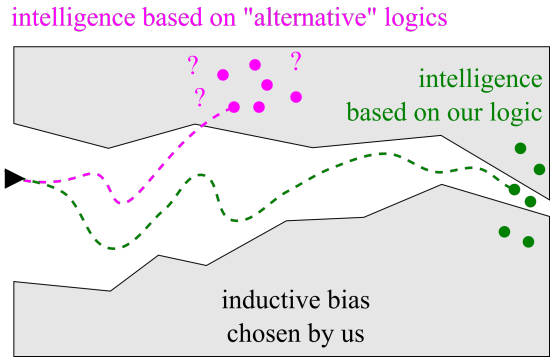
In other words, each Transformer layer takes N inputs and produces N equivariant outputs. That is the same as saying that *each* output is permutation-invariant in all its inputs. As we explained in the last section, permutation invariance is the symmetry that characterizes a logic as having *individual* propositions.

In **Multi-Head Attention**, the intermediate computations are duplicated multiple (eg, $M = 8$) times, each with their own weight matrices. From the logic point of view, this amounts to duplicating M logic rules per output. But since the next layer still expects N inputs, the M outputs are combined into one, before the next stage. Thus, from the logic point of view this merely increased the parameters *within* a single logic rule, and seems not significant to increase the power of the logic rule-base. Indeed, experimental results seem to confirm that multi-head attention is not particularly gainful towards performance.

A comment is in order here, about the choice of the word "head". In logic programming (eg Prolog), one calls the conclusion of a logic rule its "head", such as `P` in `P :- Q,R,S`. Perhaps the creators of BERT might have logic rules in mind?

4 "No Free Lunch" Theory

The following conceptual diagram illustrates the possibility that there might exist some form of logic that is drastically different from the symbolic logic currently known to humans:



(13)

[2] From blog article: Illustrated: Self-Attention – Step-by-step guide to self-attention with illustrations and code https://towardsdatascience.com/illustrated-self-attention-2d627e33b20a

but there is no efficient algorithm to find them. The permutation symmetry proposed in this paper forces our logic to be decomposable into **propositions**. Such a logical form allows a mental state to be enumerated as a list of sentences (proposition), same as the "linear" nature of human **languages**. If the AGI knowledge representation is linear (in the sequential sense) and symbolic, then it would not be far from our formulation – all these logics belong to one big family.

But could there be drastically different logics? One observes that pictures and music are not easily described by words, indeed they are 2-dimensional structures. This suggests that the brain may use **multi-dimensional** matrices of features to represent the world. Such a "logic" would be very different from sequential logic and it would be interesting and fruitful to analyze the relation between them.

5 Experiment

A simple test of the symmetric neural network, under reinforcement learning (policy gradient), has been applied to the Tic-Tac-Toe game. [3]

The state of the game is represented as a set of 9 propositions, where all propositions are initialized as "null" in the beginning. During each step of the game, a new proposition is added to the set (ie. over-writing the null propositions). Each proposition encodes who the player is, and which square (i, j) she has chosen. In other words, it is a predicate of the form: `move(player,i,j)`. The neural network takes 9 propositions as input, and outputs a new proposition; Thus it is a permutation-invariant function.

In comparison, the game state of traditional RL algorithms (eg. AlphaGo [**Silver2016**] [**Silver2017**] [**Pumperla2019**]) usually is represented as a "chessboard" vector (eg. 3×3 in Tic-Tac-Toe, 8×8 in Chess, $19 \times 19 = 361$ in Go). This state vector is the same constant length even if there are very few pieces on the chessboard. Our logic-based representation may offer some advantages over the board-vector representation, and likely induces a different "way of reasoning" about the game. [4]

In our Tic-Tac-Toe experiment, convergence of learning is observed, but the algorithm fell short of achieving the highest score (19 instead of 20), and the score displayed unstable oscillating behavior after it got near the optimal value. In comparison, the board-vector version (also using policy gradient) fails to get near the highest score after 2 hours, and it also shows oscillatory behavior. Further investigation is required, but this seems to be a promising start.

6 Conclusion and Future Directions

We described a minimal AGI with a logic that can derive one new proposition per iteration. This seems sufficient to solve simple logic problems such as Tic-Tac-Toe. As a next step, we would consider inference rules with multi-proposition conclusions. The latter seems essential to **abductive** reasoning. For example, one can deduce the concept "apple" from an array of visual features; Conversely, the idea of an "apple" could also evoke in the mind a multitude of features, such as color, texture, taste, and the facts such as that it is edible, is a fruit, and that Alan Turing died from eating a poisoned apple (a form of episodic memory recall), and so on. This many-to-many inference bears some similarity to the brain's computational mechanisms [**Rolls2016**] [**Rolls2021**] [**Boraud2020**]. The author is embarking on an abstract unifying AGI theory that makes references to (but not necessarily copying) brain mechanisms.

Acknowledgements

Thanks Ben Goertzel for suggesting that neural networks are advantageous over pure symbolic logic because they have fast learning algorithms (by gradient descent). That was at a time when "deep learning" was not yet a popular word. Thanks Dmitri Tkatch for pointing me to existing research of symmetric neural networks. Thanks Dr. 肖达 (Da Xiao) for explaining to me details of BERT.

Also thanks to the following people for invaluable discussions over many years: Ben Goertzel, Pei Wang (王培), Abram Demski, Russell Wallace, Juan Carlos Kuri Pinto, SeH, Jonathan Yan, and others. Also thanks to all the university professors and researchers in Hong Kong (especially in the math departments, and their guests), strangers who taught me things on Zhihu.com (知乎), Quora.com, and StackOverflow.

[3] Code with documentation is on GitHub: https://github.com/Cybernetic1/policy-gradient

[4] In AlphaGo and AlphaZero, the algorithm uses auxiliary "feature planes" that are also chessboard vectors, to indicate which stones have "liberty", "ko", etc.

References

[1] Hitzler and Seda. *Mathematical aspects of logic programming semantics*. CRC Press, 2011.

[2] Bart Jacobs. *Categorical logic and type theory*. Elsevier, 1999.

[3] Daniel Liberzon. *Calculus of variations and optimal control theory: a concise introduction*. Princeton Univ Press, 2012.

[4] Qi et al. "Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *CVPR* (2017). `https://arxiv.org/abs/1612.00593`.

[5] Sørensen and Urzyczyn. *Lectures on the Curry-Howard isomorphism*. Elsevier, 2006.

[6] Sutton. "Temperal credit assignment in reinforcement learning". University of Massachusetts, Amherst, 1984.

[7] King-Yin Yan. "Fuzzy-probabilistic logic for common sense reasoning". In: *Artificial general intelligence 5th international conference, LNCS 7716* (2012).

[8] Zaheer et al. "Deep Sets". In: *NIPS 2017* (2017). `https://arxiv.org/abs/1611.04500`.