Symbol Manipulation in Neural Networks

Jonas Peters

May 19, 2025

Contents

1 Introduction

The human brain is truly remarkable. It ...

The study of its functionality is more in the domain of computer science: We try to model the brain with computer algorithms, which is inspired by the idea that the human brain consists of elementary computational units, neurons, which itself can be sufficiently modeled with computers, and whose interplay creates the emergent complex behavior we see.

We structure this work by logical statements like *axiom*, *premise* and *conclusion* etc. similar to mathematical writing. I believe the chain of argument is more evident this way.

Axiom 1.1. *In theory, the human brain can be sufficiently modelled by a computer program.*

The human brain consists of a very large number of neurons, around 86 billion of them. Every neuron is connected to others via *synapses*. On average, 1,000 synapses are emanating per neuron. If we were to store each synapse by one byte of computer memory, we would need 86 trillion bytes $\approx$ 86 terabytes of storage. Sounds manageable, but how long would it take to run this program? Additionally, note that we stated that each neuron itself acts as an elementary computational unit, i.e. all neurons operate in parallel. Clearly, the complexity of modelling the brain adds up enormously.

We also must not forget that the human brain is a dynamic system which operates *in space and over time*. This is in strong contrast to basic neural networks, which operate in simple input $\mapsto$ output fashion.

Example 1.1. Given the task to multiply 96459 and 29537. We humans can use the space dimension to grab ourselves a pen and a piece of paper, and use the time dimension to repeatedly perform simple multiplications and additions in order to solve the original task. A basic neural network is incapable of that.

Thus, we should think of the human brain as a device which can *create* and *perform* instructions. Hence, we may focus on the operations it can perform as a beginning. We can think of this as a human sitting in an empty room without any auxiliary items around performing a task like adding two numbers.

Premise 1.1. *Among other things, the human brain can perform operations like adding and multiplying numbers (for a restricted domain of $\mathbb{Z}$ for example), or inflecting verbs.*

Thus, we should be able to reconstruct this behavior with our models as well. Of course, it might be the case that all the operations the human brain can perform is an emergent phenomenon of the entire brain working as a whole. However, it also could be the case that the brain has certain sub-circuitry for specific tasks like adding numbers. Assuming this is the case, we also might conclude that these circuits shouldn't be much more complex than necessary, as the brain is a product of the energy minimization of evolution.

Based on these observations, our main focus is on how models, specifically neural networks, can efficiently implement a certain subset of tasks humans are naturally good at.

2 Model Framework

We consider continuous functions $f : D \mapsto I$, where $D \subseteq \mathbb{R}^m,\ I \subseteq \mathbb{R}^n$, that are implemented by neural networks. We mostly consider feed-forward networks which implicitly define a mapping from their input neurons to their output neurons.

It might feel strange to analyze feed-forward networks (in contrast to recurrent ones), since, intuitively, the human brain operates recurrently. However, there are multiple reasons for this choice: First, shortcomings of this architectural design help us understand beneficial design patterns. Second, while the entire human brain might operate recurrently, there still might be non-recurrent sub networks. Third, it is a known result from computer science that multilayer perceptrons are universal approximators (if there aren't any parameter constraints).

Theorem 2.1 (MLPs are Universal Approximators)**.** *Given any continuous function $f : D \mapsto I$, where $D \subseteq \mathbb{R}^m,\ I \subseteq \mathbb{R}^n$, and D is compact, there exists a multilayer perceptron that defines a function $f' : D \mapsto I$ s.t.*

$$\max_{\boldsymbol{x} \in D} \|f(\boldsymbol{x}) - f'(\boldsymbol{x})\| < \epsilon$$

for any $\epsilon > 0$.

Clearly, based on the assumption we have to define an encoding of the instances of our problem domain to $\mathbb{R}^n$. This encoding is also invariant in time and space, which must not necessarily be the case in human brains. If you think of the number 1, your brain might show different activity at different times (maybe one time you are more stressed out; or you think differently of the number based on your current activity, as 1 also represents the multiplicative identity for example). Still, there might be some sub-circuitry activated when adding numbers where a certain set of neurons show similar activation patterns when adding by 1.

Now, there are rather boring mappings, like $f : \mathbb{N} \mapsto \{0, 1\}, f(n) \mapsto 1[n \text{ is even}]$. Seemingly, we humans can compute this functions over the entire set $\mathbb{N}$. However, I doubt this. If I asked you whether 985094890282377640950293869 02 is even or odd, you wouldn't have any issues. But if I asked you to repeat this number instead, most of us would fail (even if you managed to do this, what about even bigger and bigger numbers?).

Why did I claim this mapping to be boring? Because this problem can be reduced to a mapping $f' : \{0, 1, \ldots, 9\} \mapsto \{0, 1\}$ (by just considering the last digit). This mapping has a finite domain, and hence also a finite image. Such mappings can be implemented by simple input-output associations.