# Scale, abstraction, and connectionist models: On parafinite thresholds in Artificial Intelligence

by

Zachary Peck

(Under the Direction of O. Bradley Bassler)

Abstract

In this thesis, I investigate the conceptual intersection between scale, abstraction, and connectionist modeling in artificial intelligence. First, I argue that connectionist learning algorithms allow for higher levels of abstraction to dynamically emerge when parafinite thresholds are surpassed within the network. Next, I argue that such networks may be surveyable, provided we evaluate them at the appropriate level of abstraction. To demonstrate, I consider the recently released GPT-3 as a semantic model in contrast to logical semantic models in the logical inferentialist tradition. Finally, I argue that connectionist models capture the indefinite nature of higher-level abstractions, such as those that appear in natural language.

Index words: [Connectionism, Scale, Levels, Abstraction, Neural Networks, GPT-3]

# Scale, abstraction, and connectionist models:
# On parafinite thresholds in Artificial Intelligence

by

Zachary Peck

B.S., East Tennessee State University, 2015

M.A., Georgia State University, 2018

A Thesis Submitted to the Graduate Faculty of the

University of Georgia in Partial Fulfillment of the Requirements for the Degree.

Master of Science

Athens, Georgia

2021

# Scale, abstraction, and connectionist models:
# On parafinite thresholds in Artificial Intelligence

by

Zachary Peck

Major Professor: O. Bradley Bassler

Committee: Frederick Maier

Robert Schneider

Electronic Version Approved:

Ron Walcott

Dean of the Graduate School

The University of Georgia

May 2021

# Contents

# List of Figures

# Chapter 1

# Introduction

In the philosophical literature, connectionism garnered a lot of attention at the end of the twentieth century. But, for whatever reason, much of that interest has petered out as connectionist based technology has begun to flourish over the past twenty years. This may be because many philosophers believe that the essentials (both philosophical and technical) of connectionism have already been sufficiently fleshed out and that, consequently, philosophical reflection on the topic has become stale. This neglect cannot be solely attributed to the philosophers' technical ignorance of new connectionist architectures. With some exceptions, most of the popular neural network architectures of the twenty-first century were created during the heyday of philosophical discussions of connectionism (i.e., the late twentieth century). What changed in the twenty-first century was the scale of computational power (via better hardware, most notably GPUs), the scale of information sharing networks (via the world wide web), and the availability of large datasets. So, the networks grew 'deeper' and were trained much more quickly on 'big' data. Changes in the scale at which connectionist models were being realized made little difference, so the story goes, with respect to what is going on philosophically behind the scenes. It's still *just* connections and thus doesn't deserve further philosophical reflection.

But this, I think, is to entirely overlook the underlying significance of connectionism. In this thesis, I argue that what makes connectionist approaches well suited for the sort of problems they have solved (mostly) in the last decade - problems that logic-based approaches to AI failed to solve - is the capacity for

higher levels of abstraction to dynamically emerge. And the key to the possibility of this emergence, I will argue, is scale. In short, once a certain threshold in the quantity and complexity of interconnections is passed, higher levels of abstraction are likely to emerge (given an adequate learning mechanism). Thus, I argue that the central philosophical issues surrounding connectionism are questions concerning scale, levels, and emergence. Yet, the Stanford Encyclopedia of Philosophy article on connectionism [9] makes no mention of the words 'scale' and 'emergence' and cites none of the philosophical literature on levels.[1] Nevertheless, it is, I contend, no matter of philosophical insignificance that connectionism has flourished in the form of *deep* learning and *big* data.

In the process of developing this view, I argue that the process of emerging levels of abstraction made possible by both the structure of connectionist networks and the scale at which those networks are currently being realized puts us in a better position to appreciate and understand the indefinite nature of meaning. There has been a rift between some of the more interesting philosophical developments of the twentieth century (e.g., deconstruction and phenomenology) and the philosophical assumptions that inform AI research. In the spirit of repairing this rift, I argue that sufficiently scaled connectionist models (such as the recently developed GPT-3) are capable of capturing some of the ideas arising from those traditions, most notably the indefinite nature of meaning.

In the second chapter, I introduce the basics of connectionist modeling. In particular, I introduce the perceptron (or, single layer neural network) and the multi-layer, fully connected neural network. By simply scaling up, multi-layer, fully connected networks are capable of quite a bit of problem solving that has evaded logic-based AI. I also introduce one of the most popular neural negtwork architectures of the twenty-first century - the convolutional neural network (CNN). This will set the stage for a summary of Buckner's conception of transformational abstraction [8], which, I argue, can be generalized beyond the specifics of the CNN architecture. To demonstrate, I argue that the success of the recently released GPT-3 [7] (which employs a transformer architecture [43]) may also be explained in terms of transformational abstraction. Thus, some attention will also be given to the technical details of the transformer architecture.

[1] It is worth noting that this article is written by Cameron Buckner and James Garson, two philosophers whose work will be discussed in more detail in this thesis.

But this will only be to show that specifics aren't all that matters from a philosophical perspective. What is often more important is the scale of the network.

Thus, in the third chapter, I flesh out the argument that the central philosophical issues surrounding connectionism are scale, levels, and emergence. First, I discuss the diversity of conceptions of levels that may be found in the philosophical literature. Then, I propose a conception of levels in terms of abstraction that I think is sufficiently general to satisfy many of the motivations for talking about levels in the first place. In any case, it will be shown that the notion of levels of abstraction is particularly well suited for a discussion of neural networks. The primary conceptual contribution of this chapter is that of the emergence of new levels of abstraction. I argue that it is the emergence of new levels of abstraction that best explains both the technological success of connectionist modeling and its philosophical significance. And consistent with my suggestion at the end of the third chapter, I suggest that the scale of a network is perhaps the most important feature to consider when explaining the emergence of higher levels of abstraction in a neural network. Finally, I consider the difficulty of identifying the right level of abstraction.

To demonstrate this difficulty, I consider GPT-3 as a candidate inferentialist model of natural language semantics in contrast to logical inferentialist models in the fourth chapter. Garson [17] provides an excellent glimpse into what a logical inferentialist model of semantics looks like, so I will provide a brief summary of his view. He develops a model of the logical connectives in predicate logic. But the problem with his account, I argue, is that it renders the meaning of the connectives definite. I conclude that any logic-based semantic model must inevitably render meaning definite. The problem is that most of what is meaningful about natural language is indefinite, a view for which I will more definitively argue in the fifth chapter. In contrast, I introduce the notion of connectionist semantic models (GPT-3 being the most notable example to date) and argue that such models are better suited for capturing the indefinite nature of meaning.

In the fifth chapter, I provide a handful of arguments that meaning is indefinite. In particular, I argue that the source of meaning for any given natural language word or phrase must be an indefinite trace that reflects some vague yet ever expanding entanglement of the entire history of human language. While the

first half of this chapter is devoted to the indefinite nature of meaning, the second half of the chapter transitions towards the ontological consequences of this fact. Most notably, because our concepts are indefinite, they tend to obscure and conceal a clear picture of reality. I emphasize this is in the context of our conceptions of self and individual agency.

# Chapter 2

# From Perceptrons to GPT-3

In this chapter, I provide a brief introduction into the world of connectionism. In particular, I emphasize that connectionist networks are *models* of transformations and that the parameters within those models must realize, in some way or another, representations of concepts necessary to model those transformations. Specific attention is given to the CNN and transformer architectures, as these will be my primary case studies. I also introduce Buckner's [8] conception of transformational abstraction in this chapter. This will set the stage for the following chapter in which I refine Buckner's conception of abstraction by generalizing it beyond the specifics of the CNN architecture.

## 2.1 Perceptron

The simplest neural network architecture is a single output perceptron [32].[1] Basically, such a network outputs a single weighted sum of its input values. Despite its simplicity, describing the perceptron will provide us with the basic vocabulary we will need for understanding more complex neural network architectures, such as GPT-3. All neural networks consist of an array of input values (**x**), an array of weights

[1] I will henceforth refer to the single output perceptron simply as 'perceptron.' Technically, there can be perceptrons with multiple outputs. In fact, some authors even refer to multi-layer neural networks as multi-layer perceptrons. I don't think anything substantial is at stake in the choice of language we employ. So, for the purposes of this paper, I ask the reader to think of perceptrons as, specifically, single layer, single output neural nets. This simply allows for easy reference to what I take to be the simplest of neural network architectures.

(**w**), and an array of output values (**y**). In the case of a perceptron, the input array will be a vector of at least two values (why it cannot be a single value is explained below); the array of weights will be a vector with at least two values (i.e., at least one weight and one bias – again, more on that below); and the output array will simply be a single value. What distinguishes the perceptron from simple regression is the introduction of some non-linear activation function ($f$), which is precisely why the domain of neural networks (unlike regression) is non-linear. (In fact, as I will explain in more detail below, if we let $f$ be the identity function, then the perceptron really is just linear regression.) This leaves us with following equation for the perceptron:

$$\mathbf{y} = f(\mathbf{x} \cdot \mathbf{w})^2 \tag{2.1}$$

For my purposes, what is significant about the above equation is that we can think of it as a model of something in the 'real' world. In particular, it is the array of weights **w** that functions as a model. Somehow, this array of quantitative values must represent (i.e., model) some transformation in the 'real' world. And what makes neural networks special is that, in principle, they are capable of approximating any given function. As we will see, the challenge to getting a neural network to approximate a specific function is identifying an adequate architecture and the right level of complexity such that the desired function (or, transformation) may be learned. Thus, if we can identify the appropriate internal structure to use, we may be able to model any 'natural' process, including communication in natural language!

To demonstrate how perceptrons may approximate any function (i.e., any function with a single output), let's start with a relatively straightforward example - an approximation (or, model) of the logical connectives, such as $\&, \vee, \neg$, and so on [31]. We can think of such logical connectives as functions from

[2] For those familiar with neural networks, one will notice that I have not included what is often referred to as the bias, or the $y$-intercept. This is because I prefer to think of the bias as the first weight in our array of weights. This is not just a convenient notational convention - it also demonstrates more clearly that the bias is a crucial part of the network's internal model. It may be important to note the consequences of using this notation. First, the array of input values must consist of an additional value that will always be set equal to 1. This value will be multiplied by the bias in the array of weights when we apply the dot product to the two arrays. For purposes of notation, these values will always correspond to the $0^{th}$ index in their respective arrays. In other words, $w_o$ will correspond to the bias and $x_0$ will always equal 1. For those unfamiliar with neural networks, it may also be important to note that, for fully-connected networks (exactly what that means will be explained below), the number of hidden weights (more precisely, the number of hidden weights in the first layer) will always equal the number of input values.

two input values (only one input value in the case of $\neg$) to a new output value, where both the input and output values will correspond to Boolean truth values. In the case of $\neg$, the perceptron only has one input (either $0$ or $1$), and its output should be the opposite of its input (i.e., if the input is $0$, then the output should be 1; and if the input is 1, then the output should be $0$). This can easily be approximated by a perceptron by setting the bias (i.e., $w_0$) equal to $0.5$ and the only weight (i.e., $w_1$ – since there is only one input, there is only one weight) equal to $-1$.

$$y = f(\mathbf{x} \cdot \mathbf{w}) = x_0 \cdot 0.5 + x_1 \cdot -1 \tag{2.2}$$

Notice that the values in the array of weights need not be the exact values I have chosen. This demonstrates that the function is merely an approximation. There will always be a range of possible models that will adequately model the 'natural' process we are attempting to approximate. This is our first glimpse into the indefinite.

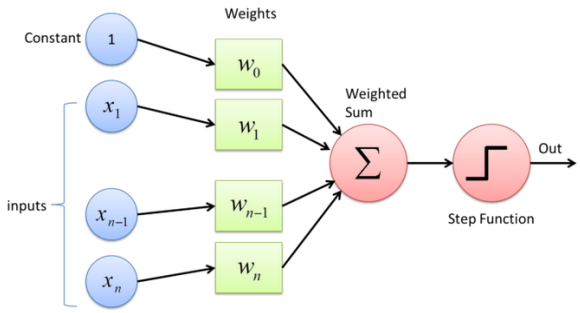
It should be noted that the above explanation is incomplete. The function $f$ was not defined. This is because, in the case of approximating $\neg$, $f$ may simply be the identify function. As I mentioned above, if we let $f$ be the identity function, then our model as a whole is simply a linear regression model. We take the leap into the world of machine learning when we make $f$ nonlinear and continuous, as will be explained below. In the case of simple logical functions, such as $\&, \vee, \rightarrow$, and $\neg$, we can actually model their behavior exactly with a discontinuous, nonlinear activation function known as a step function. Notice that, in 2.2, $y = 0.5$ when $x_1 = 0$, and $y = -0.5$ when $x_1 = 1$. This works as an approximation, but if we want to model $\neg$ exactly, we can let $f$ be the following step function, which is depicted diagrammatically in figure 2.1 borrowed from [38].

$$f(\mathbf{x} \cdot \mathbf{w}) = \begin{cases} 1, & \mathbf{x} \cdot \mathbf{w} \geq 0 \\ 0, & \mathbf{x} \cdot \mathbf{w} < 0 \end{cases} \tag{2.3}$$

You may notice that a piecewise function (such as the one above) can model the operator $\neg$ exactly without the weights. In short, if we simply input $\mathbf{x}$ (without the bias (such that it is a single value) and without

multiplying it by the array of weights) into the above function, we will get the same results. This is certainly true, but only because we are attempting to approximate a very simple function where the inputs and outputs are Boolean. As we will see below, the addition of more weights and a continuous activation function will be necessary for more complicated problems and for the process of learning to occur. The above discussion is intended to serve two purposes: 1) introduce the basics of neural networks in an accessible way; and 2) demonstrate that, even though neural networks aren't necessary for modeling logical functions, they are sufficient.

Figure 2.1: Perceptron




All of the above simply scratches the surface of what is possible when employing neural networks as models. It turns out that only very simple logical functions can be modeled using a single-layer perceptron. To model just slightly more complex logical functions, such as $\leftrightarrow$, $\oplus$ (exclusive disjunction), and so on, we must introduce a second layer of weights into the equation. This is because these sorts of functions are not linearly separable.

This fact can be demonstrated in solely logical terms. We can encode the $\oplus$ (or, $XOR$) function using only the $\&, \vee$, and $\neg$ functions: $p \oplus q = \neg(p\&q)\ \&\ (p \vee q)$. Given this logical equivalence, we should be able to see why we will need multiple layers to approximate this function. See figure 2.2 borrowed from [10] for a representation of this logical formula depicted as a network. Thus, it follows that more complicated networks that approximate nonlinear functions can be thought of as a collection of interconnected, linearly separable, perceptrons. Thus, in principle, anything a neural network can do can be accomplished by employing logic alone. Of course, this fact should be obvious insofar as, at the end of the day, most neural networks are instantiated by the logical circuitry of standard computers, which

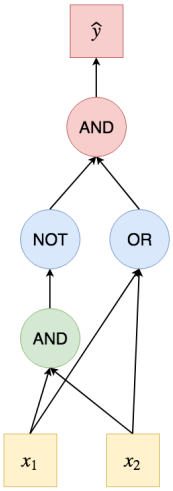



Figure 2.2: XOR Neural Net Architecture

employ simple a Boolean logic (see Neural Turing Machines for a counter-example [19]). We will return to this point in the next chapter. For now, it suffices to say that there is nothing special about neural networks with respect to the sort of functions they can approximate. Anything a neural net can do, a sufficiently complex logic program can do. The difficulty, as I will explain in more detail below, is whether or not implementing a logic program is *practically* possible.

So far we have seen how a neural network may approximate a function, but we haven't seen how the network learns to approximate those functions. The essential idea behind machine learning is error minimization. Consider the network described in 2.2. As we saw, this network will not output the exact values for which we are searching (i.e., either 0 or 1). One solution, as seen in 2.3, is to introduce a step function, but, in that case, the network hasn't really 'learned' anything. One way to get the network to learn to better approximate the desired function is to implement backpropagation [34, 35]. This is where we must introduce a continuous activation function. This will allow us to compute the partial derivative of the error with respect to each weight. Programming the network to do this allows the network to identify the contribution of each weight with respect to the overall error. Once this is computed, we can implement what is known as gradient descent. Essentially, the idea is that small changes will be made

to each weight such that, over time, the network will descend the error gradient in order to find a local minimum.

Thus, continuity of activation functions seems to be the crucial ingredient providing neural nets with a capacity to learn. In theory, neural nets can approximate any given function. The challenge is identifying the scale of parameters required to model the function and the values those parameters should take - no small challenge at all. What distinguishes neural nets from regression is simply that we have a nonlinear activation function, which makes an otherwise linear computation nonlinear. A popular activation function is sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}}$. One of the reasons this function is so popular is because its derivative is quite easy to compute programmatically: $\sigma'(x) = (x)(1 - \sigma(x))$. When we stack layers, we can identify the local effect each layer is having in its contribution to the error. As I hope becomes clearer as the thesis progresses, deeper networks allow for highly abstract, indefinite concepts to emerge. This is because these small changes across a multi-dimensional, (where the dimensionality seems to be huge in the parafinite sense - which will be explained later) continuous gradient allow for the network to identify vague and indefinite concepts that are intrinsically dynamic and capable of subtly changing from learning experience to learning experience.

However, all of the above is a description of the ideal scenario. In practice, gradient descent may never find a sufficient local minimum. It may get stuck with an error rate that is too high. In such a case, the network would have failed to identify a satisfactory model. It may also find some local minimum only to significantly increase the error in the next iteration. To address this issue, neural net engineers may need to implement mechanisms that stop the network from updating its weights in unproductive ways. But they must be careful in how such mechanisms are implemented, because it may prevent the network from finding an even smaller local minimum. Another problem is that, if the learning rate is too high, the network may simply jump around without consistently moving in one direction across the gradient. Similarly, if the learning rate is too small, then the network may never reach a local minimum in a satisfactory amount of time. The learning rate is known as a hyperparameter, i.e., parameters that do not directly contribute to the internal representation realized by the network (such as the specific values taken

by the weights) but influence either the structure of the network (e.g., the number of hidden weights) or the learning process (e.g., the learning rate). In a nutshell, my point is that engineering a network is a bit of an art form. It requires tweaking hyperparameters until the network has satisfactorily reduced the error.

## 2.2 Multi-layer, fully connected neural nets

To approximate more complicated functions, we must add more layers to our perceptron. In this section, I will present the basics of multi-layer, fully connected networks. It is in this presentation that we will be able to fully appreciate the sense in which neural nets are capable of approximating any function. Moreover, in this section, I will generalize the presentation we saw in the previous section such that, not only will the conceptual machinery for multiple layers be introduced, but I will also introduce notation for inputs and outputs of higher dimensions. Recall that, in the previous section, inputs were 1-dimensional vectors and outputs were scalars. At the most general level of abstraction, the inputs, hidden weights, and outputs of a neural network should be understood as tensors.

For the purpose of demonstration, I will describe a multi-layer neural architecture in which the input is a 2-dimensional tensor (or, matrix) and the output is a 1-dimensional tensor (or, vector). This is the the framework we would need for image classification. Images are simply 2-dimensional arrays (or, matrices) of pixel values. Perhaps the most popular example of an image classification problem is hand-written digit recognition. The popular MNIST dataset, for example, consists of 28x28 sized images of hand-written numbers, where the output of the network is a representation of the digit in the particular image input into the network. Since we will need a representation that is easily amenable to the process of gradient descent, we cannot simply represent the digit using a single value because there is no natural way to calculate the error. Instead, the output will be a vector of length 10, where each index in the vector corresponds to one of the classes, in this case one of the ten digits. So, for example, if the image input into the network is a handwritten '0', then the ideal output would be the following vector: $\{1, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$. This indicates that the network has identified the class associated with the $0^{th}$ index in our vector, in this case

the digit '0.' It should be noted, however, that this is an ideal output. In practice, we should expect these values to range from $0$ to $1$ rarely equaling exactly $0$ or $1$, and we will simply take the largest number as the network's 'choice.'

Given the complexity of this problem, we will need a more complex neural network architecture to solve it than a simple perceptron. It is certainly more complex than the $XOR$ function! While the most successful network is a CNN (which I will introduce in the next section), a 2-layer, fully connected network is capable of solving this problem with over $99\%$ accuracy. To introduce a second layer, we must introduce a second set of weights. This is precisely what an additional layer is, i.e., an additional set of weights (or, parameters). Adding an additional set of weights $W_1$ (and an additional activation function) to 2.1, we get the following formula.

$$Y = f'(f(XW_0)^T W_1)^3 \tag{2.4}$$

There a few things to note about this formula. First of all, our input is no longer a vector and is now a matrix. Consequently, our weights must also be matrices. As I said above, we should, in general, think of all the components of a network as $n$-dimensional tensors. Technically, in this case, our output is still a vector, but I have chosen to use matrix notation for consistency (a vector is, of course, a 1x$n$ matrix). Notice that, in the second layer, the matrix multiplied by our second weight matrix is the transpose of the output of our first layer (i.e., $f(XW_0)^T$). This is simply to get the output of the second layer to equal the desired dimensionality of our output (see the footnote for further clarification). Finally, one will notice that there are now two activation functions $f$ and $f'$. As we saw in the previous section, these nonlinear activation functions allow for a derivative to be calculated in the backpropagation process. Having two functions allows for a derivative to be calculated for each layer. Without a nonlinear activation function in the first layer, the two layers would 'collapse' into a single function and it would not be possible to

[3]For the interested reader, the dimensions of the above matrices are provided here. As stated above, $X$ is 28x28. Since we want $Y$ to be 1x10, $W_1$ must be $n$x10. In this case, $n$ must equal 28, since our input is 28x28. From this, it follows that $W_0$ is 28x1. And from this it follows that the output of the first layer $f(XW_0)$ is 28x1. Thus, to get the desired dimensionality of the output of the whole network, we must multiply $W_1$ by the transpose of $f(XW_0)$, which is 1x28.

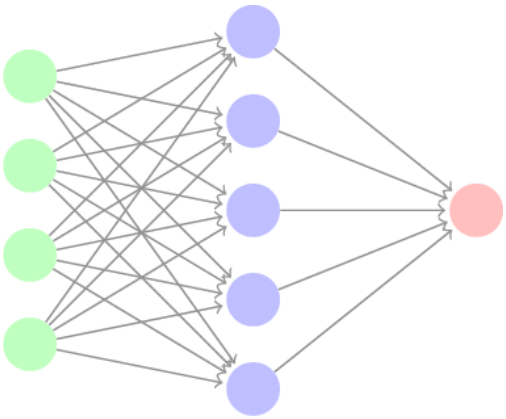


Figure 2.3: Fully Connected Network

isolate each layer's effect on the output. To be sure, there are multiple ways to present this formula, but I think this presentation will suffice for my purposes.

So far, we have seen the addition of another layer, but we haven't discussed what it means for these networks to be fully connected. In part, this is due to the fact that all of the networks we have discussed have consisted of only a single hidden unit, i.e., only a single weight matrix for each layer. Notice, however, that the entire input matrix is multiplied by the single weight matrix $W_0$ in both 2.1 and 2.4. This is essentially what is meant when we say the network is fully connected. The point becomes slightly clearer if we introduce another hidden unit, as can be seen in figure 2.3. In this case, we now have two weight matrices ($W_0^0$ and $W_0^1$), and the entire input matrix is multiplied by each of these matrices. In other words, this network is fully connected. The point should become clearer next section when we consider a non-fully connected network architecture, i.e., the CNN.

What is most important, for the purposes of this thesis, is that the addition of more layers and hidden units introduces the capacity for greater internal representation in the network. As I will explain in more detail in the following chapter, adding additional layers allows for the 'deeper' layers to instantiate a higher level of abstraction than that which may be realized by 'shallower' layers. To provide a concrete example, if the first layer has learned to detect the presence of low level features such as lines and curves, the second layer may learn to represent more complex features such as squares and circles. Furthermore, by adding

more hidden units, each layer has the capacity to detect more than one feature. For example, $W_0^0$ might learn to detect the presence of vertical lines while $W_0^1$ might learn to detect to the presence of horizontal lines. And with each hidden unit in one layer connected to each hidden unit in the next layer, higher level features that are combinations of various lower level features may be detected.

Multi-layer, fully connected networks have been successful in solving a wide variety of problems that had evaded logic-based approaches to AI. As mentioned earlier, 2-layer, fully connected networks are able to classify handwritten digits with over 99% accuracy. But this is only accomplished by significantly scaling up the number of hidden units. As reported by Lecun [26], the 2-layer network with the highest testing accuracy (specifically, $99.3\%$) on the MNIST dataset consists of $800$ hidden units. Most likely, this network has learned to detect the same features but in different locations, which is why it requires such a large number of hidden units. Hidden unit $W_0^{467}$, for example, may detect vertical lines in the bottom left-hand corner, perhaps ranging over only a handful of pixels. Scaling up the size of the fully connected network slightly improves the accuracy - of course, there's not much room for improvement when you're already above 99%! Again, as reported by Lecun, the most accurate ($99.65\%$) fully connected network with respect to the MNIST dataset consists of 6 layers, with $784$, $2500$, $1500$, $1000$, $500$, and $10$ hidden units respectively.

## 2.3 Convolutional neural nets

In contrast to fully connected networks, CNNs are better suited for detecting local features in an image (e.g., lines, curves, or shapes) that, when combined, contribute to the formation of the 'thing' (e.g., chairs, bears, or people) that the network is being trained to detect [15]. As we saw with fully connected networks, there is a sense in which the informational content of each pixel value is present for each 'step' in the network. As we will see in this section, CNNs use filters (or kernels) to focus the network's attention on specific local features by increasing their weighted significance. This allows CNNs to develop internal

representations of specific features, such that the same subset of a network's resources may be responsible for detecting that feature regardless of its position in the image.[4]

The convolutional filter is simply a matrix with a smaller dimensionality than our input image. This can be any size, but it should be big enough to detect relevant features while small enough to only attend to one feature at a time. For example, it may be reasonable to use a filter of size $3 \times 3$ for the MNIST dataset. The values that populate these filter matrices may be chosen by the engineer to detect a specific feature or may be parameters to be updated as the network learns. Typically, the latter is implemented so the network is capable of learning for itself what features are most relevant to the specific task. As was mentioned in the previous section, typical features we may want to detect at the lowest level include things like horizontal, vertical, and diagonal lines, curved edges, and so on. Because these features will be detected regardless of their position, this will significantly reduce the scale of parameters necessary for such feature detection.

Through a process referred to as downsampling, the filter outputs a single value for each region of the same size, where that value (in a sufficiently trained network) essentially signifies whether or not the feature associated with that filter was present. After spanning over the entire input, each filter produces (or, is associated with) what is referred to as a feature map. The specific dimensionality of the feature map will be determined by the size of the original image, the size of the filter, and what is known as the stride, which is simply a measure of how much the filter moves each time it slides from one position to the next. For example, if we apply a stride of size $1$ (i.e., the filter will move $1$ pixel to the right in each iteration) with a $3 \times 3$ filter to a $28 \times 28$ image, then our feature map will be $26 \times 26$. Essentially, a stride of size $1$ means that every $3 \times 3$ region of the original image will be assigned a single value in the feature map.

Understanding the informational significance of the feature map is, I think, crucial to understanding why CNNs are so successful. In essence, a feature map represents where a specific feature has been detected

[4] CNNs are not well suited for detecting features regardless of *orientation*. Generally, this can only accomplished in a CNN with additional filters (e.g., one for detecting feature X oriented horizontally, another for detecting feature X vertically). In a sense, this means CNNs really aren't capable of detecting features in a way that is orientation invariant. See Capsule Networks for an example of a neural network architecture capable of reliably detecting features regardless of both position and orientation.[36]

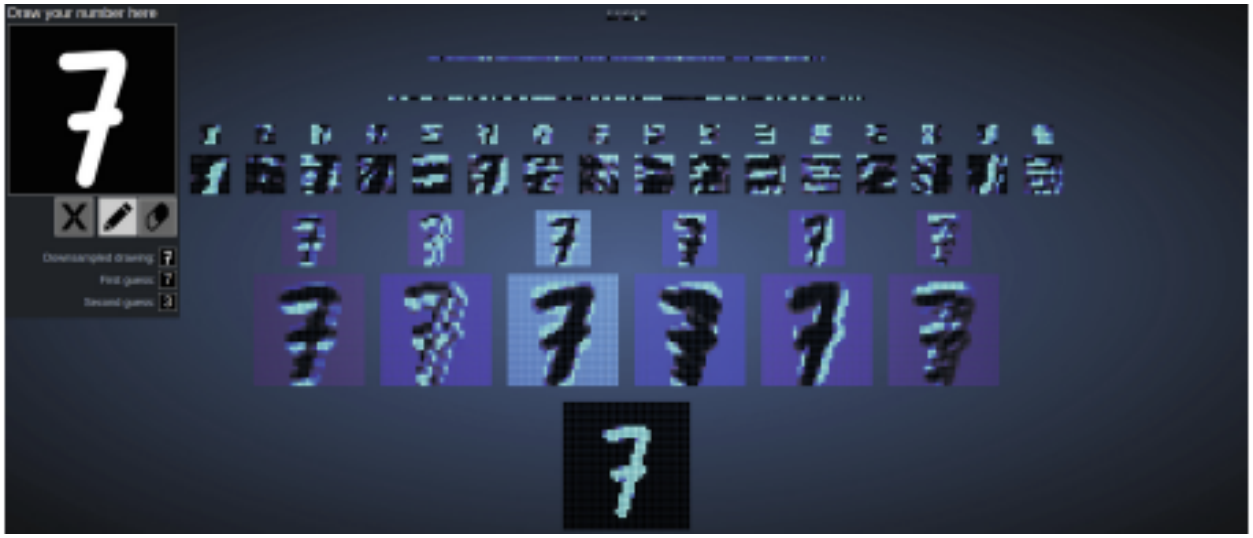



Figure 2.4: Opening up a CNN

throughout the original image. In images with higher resolution and color, a CNN may identify more subtle features. Moreover, with more complicated image classification tasks (i.e., relative to handwritten digit recognition), a CNN may identify features at higher levels of abstraction. Thus, just as we introduced more layers to the fully connected network to increase its capacity for internal representation, we may also stack multiple layers in a CNN. By performing a convolutional operation on a feature map, we may be able to detect such 'higher-level' features.

One can see a visual example of how CNNs work in Figure 2.4. The image at the bottom of the figure is the input image, which is, to us humans, clearly a handwritten number '7'. The second layer (i.e., the layer consisting of 6 images) displays the first layer of feature maps. Thus, each image in this layer corresponds to a specific filter. I think the best examples for demonstrating the point can be found in the first and fifth feature maps. The first feature map seems to be detecting horizontal edges, while the fifth feature map seems to be detecting vertical edges (or perhaps diagonal edges oriented vertically). And as we move to higher layers, higher level abstractions are identified. This will become clearer when we discuss circuits and more complex image classification tasks (e.g., facial recognition) in the next chapter.

## 2.4 Transformational abstraction

One of the key questions surrounding deep learning is exactly how and why it is so successful. One of the central problems with respect to answering this question is that the 'inside' of a neural network is typically considered to be a black box. The sheer scale of individual mathematical computations being performed within a neural network makes any sort of fine-grained understanding of how all the parts instantiate the whole epistemically unavailable. Recently, however, there have some techniques developed for opening this black box, some of which I will discuss in more detail in the following chapter. In this section, I summarize Buckner's conception of transformational abstraction, which he proposes an explanation of how neural nets (more specifically, CNNs) work. In the next chapter, I attempt to extend his conception of transformational abstraction in the next chapter to account for neural network architectures more generally.

Although Buckner aims to explain the internal workings of neural nets in [8], he is also interested in filling a lacuna in the empiricist tradition. Early modern philosophy is typically framed as a general debate between rationalists and empiricists. Concerning the formation of abstract concepts in the human mind, empiricists argue that these abstractions are learned (at least in part and perhaps entirely) from sensory experience, while rationalists insist that our minds come already equipped with some sort of framework for understanding abstract concepts. While, from an empiricist perspective, rationalists tend to undervalue the process of learning from sensory experience, empiricists in the early modern period were notoriously unable to explain how the process of learning actually occurs. As Buckner puts it, the transformation of sensory experience into abstract conceptual representations seems to be an instance of magic.

Citing [18], Buckner discusses four different accounts of abstraction that he hopes to unify in the process of explaining the magic of transformational abstraction. One account is "abstraction-as-subtraction," which suggests that abstractions are formed by removing (i.e., subtracting) those particular features that are irrelevant from the perspective of the abstract category. He associates this account of abstraction with John Locke. The second account is "abstraction-as-representation," which suggests that abstractions

are merely prototypical representations of particular objects that fall under the umbrella of the abstract category. He associates this account with George Berkeley and David Hume.

Before introducing the last two conceptions of abstraction, let's consider the problems associated with each of the above mentioned views. Locke infamously describes the "general idea of a triangle" as "neither oblique, nor rectangle, neither equilateral, equicrural, nor scalenon; but all and none of these at once" [30]. Many philosophers (including later empiricists such Berkeley and Hume) have ridiculed this account of abstraction due to its presumed incoherency. What, if anything, would a triangle without any triangular properties look like? An alternative approach (like that proposed by Berkeley and Hume) is that abstractions are simply exemplar representatives. But this simply creates another, albeit slightly different, problem - how does the mind identify an appropriate exemplar to represent some abstract category? Both accounts seem to render the process of transforming sensory experience into learned abstract representations a mystery.

The third account of abstraction Buckner considers arises out of the logical and mathematical traditions wherein abstract concepts are considered to be invariant under certain transformations - "abstraction-as-invariance." For example, the validity of a proof (which is the abstraction in this example) is invariant under permutations of the domain of the argument. Although this account works well for capturing logical and mathematical abstractions, it doesn't work well for everyday concepts, such as a chair.[5] The fourth and final account is "abstraction-as-composition," wherein abstract concepts are considered to be constructions from elementary building blocks. In the case of CNNs, we might think that they form abstractions by hierarchically constructing them out of pixel data. This account, however, will either fall victim to one of the above mentioned problems - either composite abstractions are formed without particular properties (which is, prima facie, incoherent) or a particular exemplar is formed (in which case, it's far from clear how the 'right' exemplar would be chosen).

Buckner proposes what he refers to as an "ecumenical solution." The basic idea is that an agent capable of forming abstractions must be capable of a variety of transformations. This agent must be able to hier-

[5]This is a reference to Brooks' famous argument that AI is far from realizing the sort of intelligence it takes to represent concepts such as 'chair.' [6]

Figure 2.5: Matisse's *The Back Series*

archically compose sensory experience into composite representations but must also be able to subtract particular features of representatives into the sort depictions of objects found in abstract and cubist art (Picasso being an excellent example of this sort of thing). Of course, we may never truly remove all of the particular properties, so, to some extent, representative exemplars must be used. But what's important is not so much how a specific abstract category is represented. It's the process of transforming that category from particulars into abstract composites, from abstract composites into representative exemplars, and so on. Buckner displays Mattise's *The Back Series* as an example of transformational abstraction, provided in figure 2.5. The abstract representation of a human back is not to be found in any of the four depictions; rather, it is to be found in the capacity to transform these depictions back and forth all the while maintaining an invariant conception of the human back. Thus, the abstract category 'back' is realized as a transformational capacity rather than a specific visual image or representation. I will discuss the significance of thinking about abstraction as a transformational capacity, rather than a property of something, in the next chapter.[6]

## 2.5 GPT-3

In this section, I describe the architectural details of the recently released by GPT-3 by unpacking the meaning of the acronym GPT - Generative Pretrained Transformer. As we will see, the first two terms

[6] In other words, abstraction is better understood as a process, or function, or procedure, or capacity rather than as a state, or property, attributable to some thing.

are relatively straightforward. Thus, most of the attention will be given to explaining the details of the transformer architecture, which will involve explaining the details of what has been referred to as an attention mechanism.

GPTs are generative insofar as they are designed to generate a symbol, or string of symbols, in response to some input string of symbols. In principle, all tasks involving language can be conceptualized as generative tasks. The task of writing a news article may be conceptualized as the process of generating a sequence of symbols to follow some prompt, where the prompt may be the title of the article or a description of a requested article from an editor. Conversation may be conceptualized as the process of generating a sequence of symbols to follow the string of symbols produced by one's interlocutor. Although it may seem a bit reductive, all of language production can essentially be thought of as the process of generating output strings of symbols in response to input strings of symbols. What makes the task challenging is that it is far from obvious that there is any systematic method for transforming input sequences into output sequences. In fact, given the fact that there is infinitely many distinct, yet (more or less) acceptable, sequences of symbols that may satisfy, for example, an editor's request for a news article on some specific topic, the task of language generation may seem, at best, incredibly difficult and, at worst, downright intractable. And this problem is amplified if we are attempting to create a domain-general (as opposed to a domain-specific) model of language generation that will satisfy an editor's request, a friend's question, a professor's writing assignment, and so on.

The next letter in the acronym is P, which stands for 'Pretrained.' In a nutshell, GPT-3 is pretrained on an incredibly large corpus of examples of (mostly) natural language - there is also computer code, mathematical formulas, and highly technical writing that may not be considered 'natural.' At the most general level, this simply involves training a network to predict the most probable token (where tokens roughly correspond to syllables[7]) in a sequence based upon the probabilistic relations between words observed in the training set. GPT-3 was trained on an aggregated dataset. The largest portion of this aggregate came from the CommonCrawl dataset, which is collection of text data collected from scraping

[7] See [7] for further clarification. Tokens are typically 3 to 5 characters in length and often correspond to syllables, but not always.

the world wide web that consists of nearly a trillion words. This dataset was filtered, by removing "low quality documents" and "deduplicating" documents with high overlap. Details of this process can be found in appendix A of [7]. Other datasets include WebText2 (which was also generated using web scraping techniques, but over a longer period of time), Books1 and Books2 ("two internet-based books copora"), and "English-language Wikipedia." The datasets were also given differential weights in the training process, with CommonCrawl receiving the highest weight (60%) and Wikipedia receiving the smallest weight (3%). This means that, while sequences found in the CommonCrawl dataset were observed more than once during training, some sequences from Wikipedia were not observed at all.

There are two important aspects of GPT-3's pretraining to note. First, the scale of the data on which it was trained is enormous. If our adjudicating criteria in the race for superintelligence is to create an artificially intelligent agent with above-human intelligence, we should remember that GPT-3's linguistic capacities will be, in part, due to the scale of its training data. No human alive can claim to have read as much as GPT-3 has 'read.' Second, what is most remarkable about GPT-3 is that it is designed to be general purpose. Previous natural language processing techniques typically involve pretraining a model on a large dataset (similar to GPT-3) and then *fine-tuning* that dataset for a specific task. What this means is that, when the model is applied to a specific task, it will first go through a process of updating its parameters based upon a training set specifically crafted for the task at hand. GPT-3, on the other hand, is not fine-tuned. Rather, GPT-3 is trained on the large aggregated dataset described above and is then simply given examples of the task to be completed before being asked to perform the task itself. Thus, it never updates any of its parameters as it 'learns' to perform various tasks. This is an incredible feat indicating that GPT-3 has truly realized a relatively general form of linguistic intelligence.

The title of the paper accompanying GPT-3's release is, "Language Models as Few-Shot Learners." Few-shot learning refers to the process of providing some model with a 'few' examples as a prompt for a specific task. As mentioned above, it is important to remember that this process does *not* involve any weight updates. Much of the results reported in [7] involve comparisons between GPT-3's performance given zero examples (zero-shot), one example (one-shot), and $K$ examples ($K$-shot), where $K$ is typically

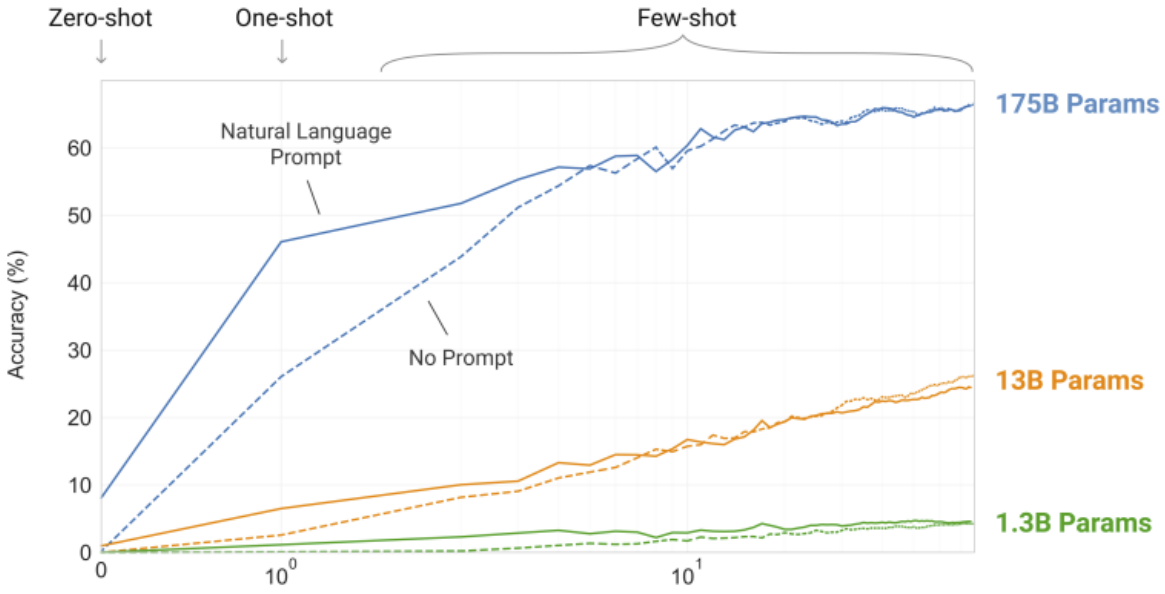



Figure 2.6: K-shot Comparison

between 10 and 100 but never greater than the model's context window which was set to 2048. As can be seen in figure 2.6 borrowed from [7], GPT-3 (which is the model consisting of 175 billion parameters[8]) makes a significant jump in performance when simply provided with a single example (as opposed to being provided with no examples) that is not seen in smaller models. This demonstrates the primary significance of GPT-3, which is that, when sufficiently scaled, a language model can realize a sort of general intelligence that allows it to perform relatively well on a novel task (i.e. a task on which it has not been trained to perform) with only a 'few' examples.

The last letter in the acronym - T - stands for transformer. A transformer is a neural network architecture designed specifically for NLP. The primary feature of the transformer is the attention mechanism, which I will now explain in some detail. Let each word correspond to a specific token $t_i \in D = \{t_1, t_2, ...t_n\}$, where $D$ is the set of all tokens in the language (i.e., the dictionary) and $n$ is the number of

[8] To be clear, all three of the differently sized models in figure 2.6 are instances of the same model and are technically smaller versions of GPT-3. But, in this thesis, I am simply following the convention of the authors of [7] in referring to the largest version as *the* GPT-3.

distinct words in the language.[9] For each token $t_i$, we assign a distinct vector $v_i$ which acts as an encoding of that token. The attention mechanism is designed to transform an input sequence of word vectors $(v_1, v_2, ...v_n)$ into an identically sized set of output vectors $(y_1, y_2, ...y_n)$ where each output vector $y_i$ can be understood as a contextualized (i.e. contextualized with respect to the input sentence) representation of the input vector $v_i$.

For example, say we have some input phrase, "bank on the river." The key to modeling the meaning of this expression is to encode the semantic relation between the words 'bank' and 'river' so that we know that 'bank' refers to land on the side of a river rather than a financial institution. Thus, we want to transform the vector encoding of the word 'bank' into a contextualized vector that contains information that will help us identify the intended meaning of the word in the expression in question. Translating our expression into its vector encoding, we get a string of vectors that, for the purposes of this example, we will denote with the following set of vectors, $\{v_1, v_2, v_3, v_4\}$. To see how the basic attention algorithm works, let's follow the transformation of $v_1$ (which is the vector we have associated with the token word 'bank') into $y_1$. First, we compute the dot product of $v_1$ and every other vector in our input string. This leaves us with four scalars, $s_{11}, s_{12}, s_{13}$, and $s_{14}$ where $s_{ij} = v_i \cdot v_j$.

But because we want to encode information about the relational significance of each word pair *relative* to the relational significance of each other word pair, we want to normalize these values such that their sum equals 1. This leaves us with the following normalized values that we will refer to as weights $(w_{11}, w_{12}, w_{13}, w_{14})$, as they indicate the weighted significance of each token with respect to the target token. Thus, $w_{14}$ should be the most significant weight insofar as it corresponds to the relational significance of the token 'river' with respect to 'bank,' which in this case is the most important relation to consider for determining the meaning of 'bank' in this context.

To generate our output, we sum the dot product of each normalized weight $w_{1j}$ and its corresponding word vector $v_j$ in our input string and sum the outputs. This leaves us with an output vector $y_1$ that we can think of as the contextualized (i.e., contextualized with respect to the rest of the sentence) word vector

[9] Recall that, in GPT-3, tokens are actually sub-word components. For the sake of simplicity, I will introduce the transformer architecture by treating tokens as complete words.

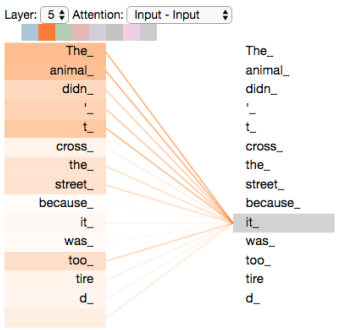



Figure 2.7: Transformer Illustration

corresponding to the original word vector $v_1$ (which, in our example, is simply an arbitrary encoding of the word 'bank'). We then repeat this process for each word vector in our input string generating the output array of contextualized word vectors, $\{y_1, y_2, y_3, y_4\}$.

In Figure 2.7 borrowed from [1], we can see an illustration of the weighted significance of the relation between the word 'it' and every other word in the sentence. The strongest connections are between 'it' and 'the' and 'animal.' The weights of these connections are essentially encodings of the contextualized meaning of the words in a sentence.

Notice that each input vector is 'used' in this algorithm three times. For the purposes of this explanation and our general trek towards a higher level of abstraction, it is helpful to think of each instance of the input sequence as its own matrix, where the number of columns is equal to the dimensionality of our word vector encoding and the number of rows is equal to the number of tokens in our input sentence. A database analogy is used to keep track of each of these matrices: keys $K$, queries $Q$, and values $V$.[10] In the above paragraph, we focused on the algorithm from the perspective of a single row in the query matrix $Q$, where that row corresponds to the word whose context we are attempting to derive. In figure 2.8 borrowed from [43], we see this process more generally where each word's context is being calculated simultaneously.

[10]It is a general goal of this thesis to pay close attention to the metaphorical and analogical language employed by AI researchers. From what I can tell, this database analogy is a very rough analogy. Besides noting that the query essentially acts as the word whose context is being 'queried,' I don't think there is much explanatory value to this analogy (unless, of course, one already has a solid understanding of databases. So, I omit an attempt to provide an explanation.

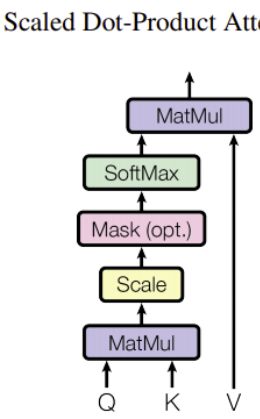



Figure 2.8: Basic Attention Mechanism

We can see that the $Q$ and $K$ matrices are multiplied. This creates a matrix with dimensionality of $n \times n$, where $n$ is the number of tokens in the input sequence. (The mathematically inclined reader may notice that, to get this output, we are technically multiplying $Q$ by the transpose of $K$ as seen in equation 2.5.)

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.5}$$

This network is referred to as an attention mechanism insofar as it identifies the words in some input sentence on which the network should be focused in order to understand the meaning of a specific word in the sentence. And by doing this for every word, as we saw above, the mechanism is able to extract a more contextual representation of each word. To be more precise, researchers often refer to the above as a *self*-attention mechanism. In general, an attention mechanism transforms noisy time series data into a continuous line that doesn't overfit the data. This may be done by re-weighting the data by projecting it onto some continuous distribution. In the case of self-attention, the re-weighting process involves a sort of self-reference to the data itself by considering the strength of the connections between the tokens in the input data. Personally, I choose to ignore this nomenclature (with the exception of this brief clarification) as I find the metaphor misleading. The thing in question that is attending to itself is the input data, not

the attention mechanism. Thus, if we are thinking of the network as an agent, then referring to this mechanism as a self-attention mechanism may mislead us into thinking that the agent (i.e., the attention mechanism) is attending to itself. In any case, what people have chosen to call a particular neural net architecture is generally not important unless we are attempting to understand the underlying metaphor and its relationship to the social mythology of that field of researchers. In this case, my point is merely that it is primarily attention (and not self-attention in particular) that deserves our attention as the underlying metaphor.

The key to the representational capacity of the attention mechanism is the parameters associated with each of the input matrices $Q$, $K$, and $V$. In diagram 2.8, notice that each of these matrices is input into a MatMul (i.e. matrix multiplication) cell. For each input matrix $Q$, $K$, and $V$, there is a unique matrix of weights that can be updated through backpropagation. What this allows is for the network to learn three different representational frameworks that operate in unison to extract contextual information about the semantic relations between words. Moreover, we can run multiple attention mechanisms in parallel, as demonstrated in 2.9 borrowed from [43], to increase the scale of distinct representational frameworks that the network might learn. As we saw with CNNs, having a multiplicity of distinct filters increases the representational capacity of the network. Similarly, adding multiple attention mechanisms increases the representational capacity of the network. While distinct filters may learn to identify distinct local visual patterns (such as curves, edges, lines, etc.), distinct attention mechanisms can learn to identify distinct global linguistic patterns (such as noun-verb relations, noun-pronoun relations, and so on).

Now we should be in a position to see how the attention mechanism instantiates a sort of transformational abstraction. The differences between the two architectures (i.e. CNNs and attention mechanisms) lies in the sort of relations the networks are designed to recognize in the input data. Convolutional filters focus on relations between local data points. Global relations are processed only after the network has transformed local relations into abstractions. This is what makes CNNs particularly well suited to visual tasks. Attention mechanisms, on the other hand, build abstract representations of single input variables (in the case of natural language processing, linguistic tokens (whether those be syllables, words, or sen-

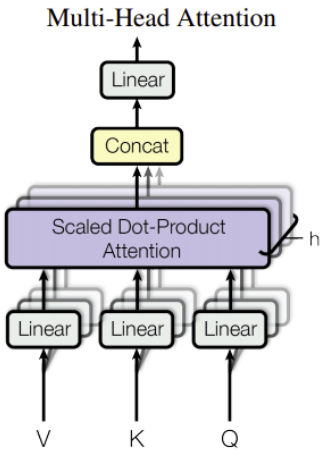



Figure 2.9: Multi-headed Attention Mechanism

tences)) based upon the global relations between that variable and all other variables in the input. This makes this network architecture well suited for learning linguistic abstractions given the variable-length and sequential nature of natural language. Whereas visual abstraction primarily involves building composites from patterns near one another in the visual field, linguistic abstraction has a reticular, weblike structure. This is, of course, to offer up a visual metaphor of a non-visual form of abstraction. I think that, for humans, reliance on visual metaphors is epistemically necessary insofar as our phenomenological world seems to be primarily constructed through vision. This, I think, explains why Buckner has identified CNNs as the prototypical mechanization of abstraction. Yet although it may be a good prototype, it is not the only form a mechanization of abstraction may take.

Before explaining the transformer architecture, I think it is first worth reviewing the disadvantages of Recurrent Neural Networks (RNNs) and CNNs with respect to sequence-to-sequence (Seq2Seq) modeling, where a Seq2Seq task simply involves transforming one sequence into another. Whereas visual information is easily reduced to a fixed size via 2-dimensional, pixelated images, natural language is not - sentences, paragraphs, and entire documents vary considerably in their relative sizes. CNNs require a fixed input size; thus, CNNs are well suited for image classification when all of the images are the same

scale. Likewise, CNNs can do quite well on natural language tasks if the input size is fixed. But given variable-length natural language inputs, this can only be accomplished by artificially padding input strings to include null values for all strings shorter than the longest string in the dataset. This is not ideal and leads to inefficient data processing.

RNNs, on the other hand, are designed to handle variable-length inputs. Consequently, RNNs were the most commonly implemented neural network in NLP until the transformer came along. RNNs are capable of processing variable-length sequential data via recursion. The problem is that the weighted significance of tokens in a sequence decreases as the network propagates forward. At each recursive step, the most recently encountered information is weighted more significantly than earlier encountered information. In other words, RNN architecture doesn't remember important information over the span of lengthy sequences. Variants of the standard RNN architecture have been developed to account for this shortcoming. The most notable is the Long Short Term Memory (LSTM) network architecture. Essentially, LSTMs include modules designed to retain significant information, i.e., they have a 'memory' mechanism.

Perhaps what is most significant about the attention architecture is that the network can outperform variants of RNNs (including LSTM) without the recursive structure. By creating a contextualized encoding of a sequence, a transformer is essentially capable of attending to the significance of relations between tokens no matter where they are located in a sequence. When GPT-2 (i.e., GPT-3's smaller twin[11]) was released, perhaps its most surprising capacity was its ability to maintain a coherent narrative across multiple paragraphs. Indeed, what is perhaps most surprising about GPT-3 is that its capacity for attending to and crafting a coherent narrative parallels with, and in some ways, exceeds our own.

In Figure 2.10, we can see the full transformer architecture. The architecture is best explained in the context of translation, one of the many natural language tasks to which the general algorithm may be applied. Say the network is translating the sentence "the snow is white" into French. Since it will need to keep track of what it is has translated so far, it needs to attend to the previous output during each step of

[11] GPT-2 and GPT-3 have the same architecture. The only difference is that GPT-2 has 1.5 billion parameters, while GPT-3 has 175 billion parameters.

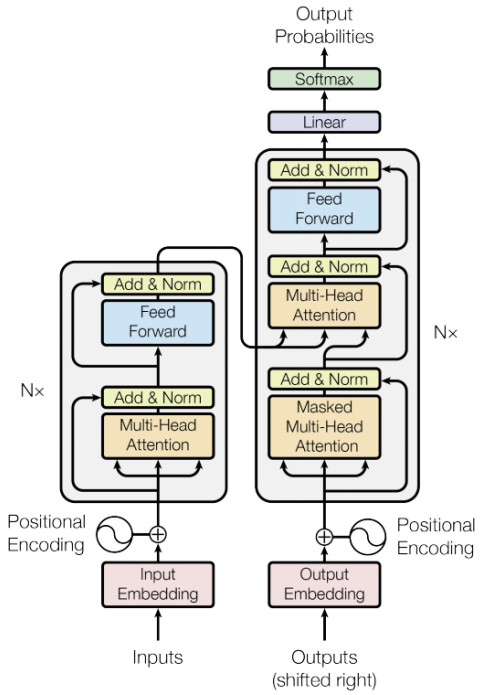



Figure 2.10: Transformer

the process. So, for example, let's say it has output "la neige est" so far. The left side encodes the original input sequence, which becomes input on the right side. The right side encodes the output so far (i.e., "la neige est") and attends to it in conjunction with the original input sentence it received from the left side. These combined attention mechanisms will most likely cause the network to attend to the words "white," "snow," and "neige" (in what seems to me to be the likely order of significance, where "white" is the most significant). Then, the network will (likely) output "blanche," which will be input back into the network. At this point, a network that is well trained on English to French translation will realize it has finished and will terminate.

What is most important about understanding GPT-3 (and transformers more generally) for my purposes is the multi-headed attention mechanism. The attention mechanism was described in detail above. Essentially, a multi-headed attention mechanism simply includes multiple attention mechanisms running in parallel. This allows the network to attend to the relational significance between tokens across a

multiplicity of different levels of abstraction. And in light of Buckner's conception of transformational abstraction, we can see that the network's capacity to transform input sequences into a multiplicity of contextualized encodings of those sequences and transform those encodings into novel, yet meaningful, output (or, response) sequences is simply a description of the network's capacity to engage in linguistic abstraction.

# Chapter 3

# Emergent abstractions and representational thickets

In this chapter, I further explore the relationship between abstraction and connectionist modeling. I begin with a formal treatment of the concept of levels (drawing from List's logical framework [27]) that, I think, acts a handy framework within which to think about levels of abstraction. The framework of levels is used to articulate an account of the emergence of abstractions. I suggest that abstractions, similar to higher-level causal processes, emerge when parafinite thresholds of complexity are surpassed in a system. With this in mind, I argue that trying to understand a connectionist network at the level of the logical relations between its parameters is akin to trying to understand the behavior of a human at the level of atomic physics - it's simply the wrong level of abstraction. Finally, I conclude by considering Wimsatt's [44] account of levels and causal thickets. I suggest that Buckner's conception of transformational abstraction may be further clarified by understanding abstractions as representational thickets.

## 3.1 Levels

There doesn't seem to be any consensus in the philosophical literature on what levels are or how liberal we should be with levels metaphor.[1] Distinct accounts of how to apply the levels metaphor abound. In general, it has become quite common to think of the standard sciences as consisting of compositional levels. At the base, we have elementary particles (physics), then atoms (chemistry), molecules (organic chemistry), cells (cellular biology), multicellular living organisms (biology), and then social groups (psychology and sociology).

I will start by summarizing a logical framework for thinking about levels proposed by List [27]. He conceptualizes each level as its own system of logic, with supervenience mappings between the propositions of each of these systems. He formally defines a system of levels as a pair $\langle \mathcal{L}, \mathcal{S} \rangle$, where $\mathcal{L}$ is a class of objects ($L, L', ...$), which are called levels, and $\mathcal{S}$ is a class of mappings ($\sigma, \sigma', ...$) between those levels, which are called supervenience mappings. [27] The system has three defining conditions. First, $\mathcal{S}$ is closed under composition, which is associative. Thus, if $\sigma : L \to L'$ and $\sigma' : L'L''$, then the composite mapping $\sigma \circ \sigma' : L \to L''$ must also be in $\mathcal{S}$. Second, each level has an identity mapping ($1_L$). Third, there is at most one mapping between any two levels. Essentially, the pair $\langle \mathcal{L}, \mathcal{S} \rangle$ form what is referred to as a category, where levels are the objects of the category and supervenience mappings are the morphisms. The first two conditions ensure that $\langle \mathcal{L}, \mathcal{S} \rangle$ is a category, while the third condition ensures that $\mathcal{L}, \mathcal{S} \rangle$ is a posetal category (which is essentially the category-theoretic analogue of a preordered set).

This logical framework fits nicely with most applications of the levels metaphor. List [27] discusses four instances of the levels metaphor that can be made consistent with one another from within the categorical framework he provides. The four conceptions are levels of granularity, ontological levels, levels of description, and levels of dynamics. Basically, the idea is that levels move from finer-grained descriptions of dynamic processes to coarser-grained descriptions. He provides the example of a coin flip. We could model the coin flip at a very fine-grained level in which we describe the atomic interactions between

[1] I am following Craver [11] in thinking of the general use of 'levels' in philosophy (most notably philosophy of science and metaphysics) as a metaphorical use of language.

the coin and its environment. Assuming we had the computational resources to do so, this provide us with near perfect predictive power. But, unless we flip the coin in an extremely controlled environment, constructing such a model is simply not feasible. On the other hand, we could move to a much coarser-grained level and simply describe the state of affairs as consisting of two equally probably outcomes, namely heads or tails. Both are possible ways of modelling the dynamic process, each with their own advantages and disadvantages.[2]

What I believe unites all of these conceptions of levels is abstraction. Of course, a likely objection is that my proposal is too subjective and strips the concept of levels of its ontological significance. I will acknowledge that I may be taking an anti-realist stance on the levels metaphor, but I would argue that such a perspective may be necessary to really get a grasp on what unites these diverse applications of the metaphor. In particular, even if there really are ontological levels, we cannot avoid the fact that, along the way to developing an understanding of those levels that are 'out there in the world,' as it were, we must first have some sort of cognitive framework in which the metaphorical conception of levels first develops. Of course, that metaphorical framework may tend towards a closer approximation of the actual leveled structure of the world over time. But the point remains that we must first have some sort of metaphorical conception, and it is that metaphorical conception that I aim to introduce in this section. Thus, my proposal need not be anti-realist and may be more akin to a structural realist suggestion. In any case, identifying where my proposal best fits in this philosophical debate is beyond the scope of this thesis. What is important for my purposes is that, as I will argue below, the technological success and philosophical significance of neural nets is best understood in terms of levels of abstraction.

One of the central reasons List, and most other philosophers for that matter, are interested in the levels metaphor is due a general interest in emergence, multiple realization, and the autonomy of higher level sciences. For example, List has argued that, because higher-level states (such as the psychological state of being in pain) seem to be realized by a multiplicity of distinct lower level states, lower-level determinism

[2] To be clear, List does emphasize that one need not be a realist about the ontological status of levels to employ his system. For example, one may only want to talk about levels of description in science within an anti-realist framework. List, on the other hand, seems to favor the unification of these four conceptions of levels and thus is presumably some sort of realist about the ontological status of levels.

need not imply higher-level determinism. To say that a higher-level state, such as pain, is multiply realized is to say that there is not one unique physical state that realizes all instances of pain. Rather, just as there is a multiplicity of possible trajectories a coin may take to land heads, there is a cluster of similar physical states that each realize the mental state of being in pain. Thus, simply because a physical level description of a coin toss, or how a person will respond to being in pain, is deterministic, it does not follow that a coarser-grained description of the process will also be deterministic; in fact, in most cases, it will not be. List [28] uses this framework to defend a compatibilist conception of free will. This argument is beyond the scope of this paper. What is important for my purposes is that higher-level descriptions reduce the quantity of information needed to describe some process by attending to coarser-grained states that are multiply realized.

Thus, returning to the language of abstraction, we can see that the move from a lower level to a higher level is essentially what abstraction is. As we saw in the summary of Buckner [8], abstraction can be conceptualized in terms of the subtraction of irrelevant features, the composition of parts into wholes, the representation of a multiplicity of stuff as one, or the invariance encountered when moving from one instance of a concept to another. Like List's unification of different ways of thinking of the levels metaphor, Buckner also attempts to unify these conceptions of abstraction in his proposal of transformational abstraction. I think both authors are, in some sense, right to propose a unification of their respective concepts (rather than a defense of conception X in opposition to Y). And this is because I take the proposed conceptual unifications to be two sides of the same coin. Namely, the move from one level to another is the process of abstraction itself. We are essentially subtracting lower-level features that may be considered irrelevant from a higher-level perspective due to the fact that those lower-level features are multiply realized by composite states described at the higher level. We can't really think about levels without abstraction, and we can't think of abstraction without the levels metaphor. To move from a lower level to a higher level is to abstract away from the details at the lower level to focus on what we might call a higher-level of abstraction.

Although a more thorough philosophical defense of the intrinsic connection between the levels metaphor and abstraction is beyond the scope of this paper, some conceptual issues should be addressed. Most notably, there is certainly a relevant and interesting distinction between red as an abstract color multiply realized by a range of color frequencies and society as an abstract entity composed of a multiplicity of individual humans. Thus, one might argue that levels of ontological granularity are distinct from levels of conceptual abstraction. This is certainly true. But this is only to point out that there is not a single category of levels that captures all instances of the levels metaphor. It does not undermine what I take to be the fact that abstraction is the defining feature of the move from one to level to another regardless of the category of levels being considered.

What's important, for my purposes, is the relationship between two adjacent levels. In particular, how do we know when we should move to a higher-level of abstraction. List is adamant that propositions, descriptions, probabilities, and causal processes are all level-specific [29]. He argues that, to reduce to higher-level states to lower-level states, there must be some proposition at the lower level capable of describing the higher-level state with finitely man terms. But, because there are uncountably many subsets of the (countably) infinite collection of possible lower-level world (he simply assumes there are infinitely many possible worlds at any given level), it follows that there are more descriptions that cannot be finitely described at the lower-level than can be (specifically, uncountably more). Thus, it is 'more likely' than not that higher-level states supervene upon some subset of lower-level states that permits no finite description at the lower-level. In fact, he suggests that it would be a "cosmic coincidence" if higher-level states were reducible to lower-level subsets that permitted finite descriptions [27]. Following List, I will refer to this argument as the combinatorial argument.

Ultimately, this argument relies on some heavy set-theoretic lifting. Bassler, for example, questions whether or not the finite / infinite distinction underlying basic set theory is philosophically tenable. In particular, he suggests that the distinction does not permit a clear cut demarcation and that a philosophical take on scale that doesn't reduce differences in scale to mere subjective relativity needs to be developed. For something that is sufficiently 'large' or 'small' that has surpassed finite description but is not necessarily

infinite, he proposes the term 'parafinite' [2]. In what follows, I will employ this term, but this is not intended to signify perfect agreement between my use of the term and his own.

With this framework in mind, we need not think of higher-level states described at a lower level as infinite in the traditional sense. Rather, we simply have to think of them as 'too large.' The obvious question is, too large with respect to what? We might think that it's too large to provide explanatory value. This, one might think, puts us back into a solely epistemic or subjectivist account of levels - it would be strange to talk about explanatory value in the context of ontological levels. I think this is on the right track, but I'm inclined to say something more ontological. I'd like to say that the description is too large to adequately capture the thing being described itself. The explanatory fact that we need higher levels of abstraction to understand the world is not necessarily a reflection of our epistemic limitations but is a reflection of the structure of the world. And, in particular, it is the fact that the world exists at a multiplicity of different scales. Garcia-Morales, for example, argues that, in order to unify physics, we must appreciate the absolute (i.e., non-relative) difference in scale between the classical and quantum levels of reality. And he proposes an account that seeks to minimize the radix economy (i.e. digit capacity) relative to the scale of the objects being described. [16]. I cite this argument not to dive into the complexities of absolute accounts of scale, unified theories of physics, ontological questions concerning the status of levels, and so on. These are quite complex issues that are beyond the scope of this thesis. Rather, my intention is only to insist that scale is far from philosophically insignificant and may, in fact, be quite important for our understanding of what abstractions are and how they relate to the 'real' world.

## 3.2 Emerging abstractions

I now turn towards an exposition of why scale is important for understanding the capacity of transformational abstraction exhibited by neural nets. The levels metaphor crucially relies on there being higher and lower levels. The movement from one level to another involves passing some sort of threshold in scale (which I refer to as a parafinite threshold) such that there is too much information at the lower level to account for the patterns being exhibited at the higher level. This proposal crucially relies on an

information-theoretic take on the process of an agent formulating an understanding their environment. The proposal is intrinsically evolutionary. As an agent attempts to model some pattern observed in their environment, it is possible (in fact, likely) that the available sensory information will be too much to adequately construct a model. Once this happens, the agent will either fail to adequately model the target phenomena, or the agent will 'evolve' such that a higher level of abstraction emerges within their internal model. It is this process of emergent levels of abstraction that allows for neural networks to solve the sorts of problems they are good at solving. And this is precisely why the scale of the most successful neural nets is far from philosophically insignificant.

What is now becoming one of the most overused description of neural nets is that they are "black boxes." This description has become so commonplace that many authors simply state it as a well known fact. Essentially, the idea is that neural nets approximate functions that seem to require internal representations at various levels of abstraction, but that the internal mechanics of how the network realizes those representations is simply unavailable to us. This, however, is simply not true. Recent methods (to be discussed in more detail below) have certainly been successful (to varying degrees) in explaining how neural nets form these internal representations. The 'box' may not be entirely transparent, but it is certainly not entirely opaque either. In fact, I would argue that it is closer to being transparent than opaque.

The idea that neural nets are black boxes is perpetuated (despite being more false than true (that is, if I'm right)) because it is, I think, part of the mythological allure of machine learning and the underlying mythology of intelligence in general. Machine learning is exciting because it is mysterious. But, if neural nets aren't really black boxes, then it's not really that mysterious. Moreover, there is a commonly held belief that the intelligence of human intuition is simply not something that can be modeled or replicated. Of all the implausible aspects of *Star Trek*, one of the funniest (in light of recent advances in AI) is a scene in which the superintelligent Data, an android capable of self-awareness, incredibly efficient communication, and so on, loses a game of chess to Deanna Troi, a humanoid. Troi suggests that there are just some things an android can't replicate, such as the 'intuition' it requires to see such game-winning moves in a game of

chess. Given the recent success of AlphaGo, the idea that human intuition is itself an irreplicable black box seems highly suspect. But the mythology of human intelligence continues as we come closer to replicating said intelligence in deep neural nets. The new extension of this age-old myth is that we will never be able to fully understand how intelligence works, even if that intelligence is something we have artificially created.

To be clear, I am not suggesting that neural nets were first considered black boxes for mythological reasons. Rather, at first, we really didn't have the right methods for looking inside the networks, and so they really were black (or, at least, dark grey) boxes. I am only suggesting that the perpetuation of thinking of neural nets as black boxes, despite advances in our understanding of how they work, is itself a feature of our mythologizing of intelligence.

One of the first attempts to look inside neural nets led to what is now known as deep dreaming, a method originally intended to open the black box of CNNs. What's sociologically fascinating about this is that, as evidenced by the nomenclature, popular reception of this technology has tended to add to the mythology that higher forms of intelligence are inexplicable. The original intention was to reveal the internal process of abstraction; instead, popular imagination ran wild by suggesting that computers were dreaming. However, if Hoel is right [22], then this association is not problematic. But this is only because dreaming may not be as inexplicable and mysterious as we tend to think. According to Hoel, dreams may be our brains' way of minimizing overfitting, which is essentially what happens when an agent fails to sufficiently generalize. A famous example of overfitting is a machine learning algorithm that has learned to distinguish between wolves and dogs simply because wolves tend to be in environments with snow. But this method of distinguishing the two types of animals fails to generalize to cases in which wolves do not appear in snowy environments. In this example, we can see that a crucial part of overfitting is the failure to develop the *appropriate internal representations* - in this case, presence of a snowy background is not the appropriate internal representation to form. Thus, the fact that an attempt to open the black box to see what sort of representations are being formed uncovered dream like images should not be surprising nor should it be considered evidence of the opaqueness of the network. In fact, quite to the contrary, this should help us understand what's going on inside.

(a) Deep Dream of Donald Trump

(b) Deep Dream of an Antelope

Figure 3.1: Deep Dream Images

Consider, for example, the images in figure 3.1. These sorts of images have become popular because of their dream like and psychedelic aesthetic. But these images actually can explain quite a bit about what's going on inside of the network. Recall that a neural network essentially updates its own internal weights by calculating the error between the actual output and the desired output and then slightly shifting the weights in a direction that will minimize said error the next time the network is given that same input. In the deep dream images, instead of changing the values of the weights, the network changes the pixel values in the input image such that the new image is more likely to be classified in some specifiable way if input into the network again. The intention of the creator of this method, Alexander Mordvintsev, was to see how the network was representing the classes it was trained to identify. But because deep CNNs include multiple layers of feature detection, we can employ this method at different levels of abstraction. The image of Donald Trump in figure 3.1a shows what happens when this process is employed at a relatively high level of abstraction. Notice that high level features, mostly facial features, are being projected onto the original image of Trump. Contrast that with the image of the antelope in figure 3.1b, where relatively low level features are being projected onto the image, mostly curves. These sorts of images provide evidence that CNNs are able to perform well at image classification by learning to internally represent features at a variety of levels of abstraction.

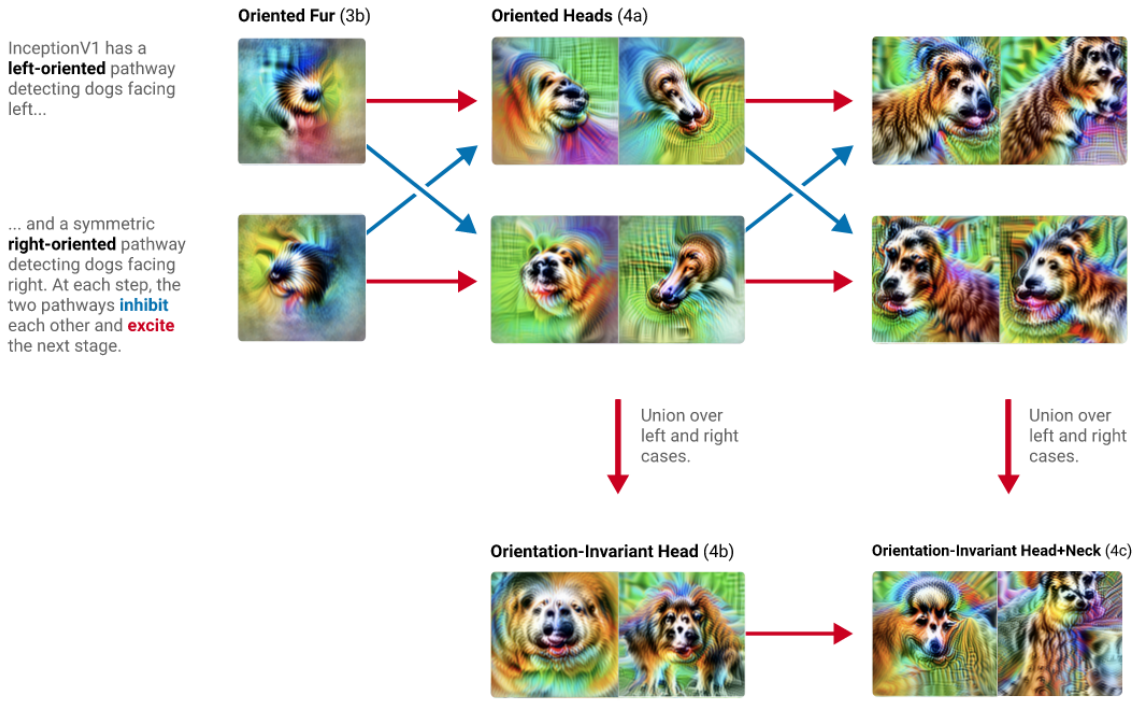



Figure 3.2: Circuitry of InceptionV1

Extensions of this technique have been abundant. Recently, OpenAI has published an article suggesting that CNNs develop internal circuits, which transform lower-level features into higher-level features [33]. Figure **??** shows an example of a high level circuit borrowed from [33]. This network presented is the InceptionV1 model trained on the ImageNet dataset [40]. InceptionV1 is trained to classify images, much of which requires the classification of different animals found within those images. In the first layer depicted, there are two distinct nodes each detecting the face of a dog with different orientations. Then, these nodes are fed to the next layer, which detects the presence of heads from different orientations. The next layers are quite interesting. They take the union of the previous layers thereby allowing them to detect the presence of heads and necks in a way that is orientation-invariant. This is another excellent example of how abstractions of the sort Locke envisioned are actually formed. These abstract representations emerge within neural circuits only to be expressed in dreams, imaginative reflection, and so on.

In the case of transformers, like GPT-3, we have a slightly different kind of emerging abstraction, but emerging abstractions nonetheless. As I intimated in the previous chapter, the sort of abstraction that emerges is both relational (i.e., it is concerned with the significance of relations between words), contextual (i.e., it is concerned with the relations between words in the specific context of the input string), and global (i.e., the relations of significance may be between any given pair tokens in a sequence, no matter how far apart they are). Given the multi-headed attention mechanism, which allows for multiple levels of relational significance to be operative in a transformer, there is likely something quite similar to circuits to be revealed in models such as GPT-3. Such a technique has not currently been developed (to my knowledge) but, I imagine, will be soon.

## 3.3 Identifying the right level of abstraction

At this point, I want to return to the idea that neural nets are black boxes. Besides the mythology of intelligence, the other primary reason why the myth of neural nets as black boxes continues is due to the common tendency (among humans that is) to think at the wrong level of abstraction. This is perhaps the primary motivation behind the literature on multiple realization and, as Fodor put it, the "autonomy of the special sciences" [13]. Physical reductionist thought has convinced many among us that, even if it is not practically possible, an atomic description of, for example, the human body must be the best possible explanation of human behavior, i.e., best insofar as it provides greatest predictive power. Some philosophers, however, (such as Fodor and List) have argued that atomic descriptions of higher-level processes is not only practically problematic but also conceptually problematic. Atomic level descriptions are simply the wrong level of abstraction at which to explain human behavior.

The best argument for this view (in my view) brings us back to the notion of invariance, in this case invariance under causal intervention. Say we want to change the behavior of some human agent. Suppose we try to intervene at the atomic level. Not only is this a practically problematic way of going about changing the agent's behavior, the scale of possible atomic changes that we might make that would have *no* influence whatsoever is huge. Moreover, it's far from clear that there would be any general property (at

the atomic level, that is) associated with those atomic changes that would invariably change the agent's behavior in the desired way. If we wanted to identify what all those atomic level interventions have in common, there is good reason to believe that we would have to appeal to higher-level psychological properties. There's simply too much variation at the atomic level to systematically and reliably explain human behavior. The sort of intervention that would invariably lead to the same change in a human agent's behavior would likely be a psychological, cognitive, or neural intervention.

This sort of insight is not only relevant to the computational sciences, it is one of the driving forces behind the creation of new programming languages. Constructing a neural network in an assembly language is certainly possible - it's done every time a neural network is constructed. But humans aren't creating deep neural nets at this level of abstraction. Rather, we write much higher-level instructions that are translated in machine code. But we don't do this just because it saves time writing the initial program. Perhaps much more importantly, it saves time if we ever want to intervene and make some sort of general change in a way that is reliable and consistent across a multiplicity of cases. Of course, at the end of the day, any program we create could, in principle, have been written in an assembly language.

Identifying the 'right' level of abstraction at which to model a process is not always easy, however. But the takeaway point is, more often than not, if we are having trouble explaining or understanding some phenomenon from within one conceptual paradigm, it may be useful to switch paradigms and think at a different level of abstraction. In the previous chapter, I attempted to demonstrate that understanding the inner workings of a connectionist models requires identifying the right level of abstraction. And in the previous section, I suggested that the view that networks are black boxes tends to assume that the right level of abstraction is to be found at computational level.

## 3.4 Representational thickets

Before moving to the next chapter, I want to review a much different take on levels than is provided by List. Wimsatt provides a much more nuanced conception of levels by focusing on the complexity of level demarcation, particularly within the life sciences broadly construed. He suggests that the causal networks

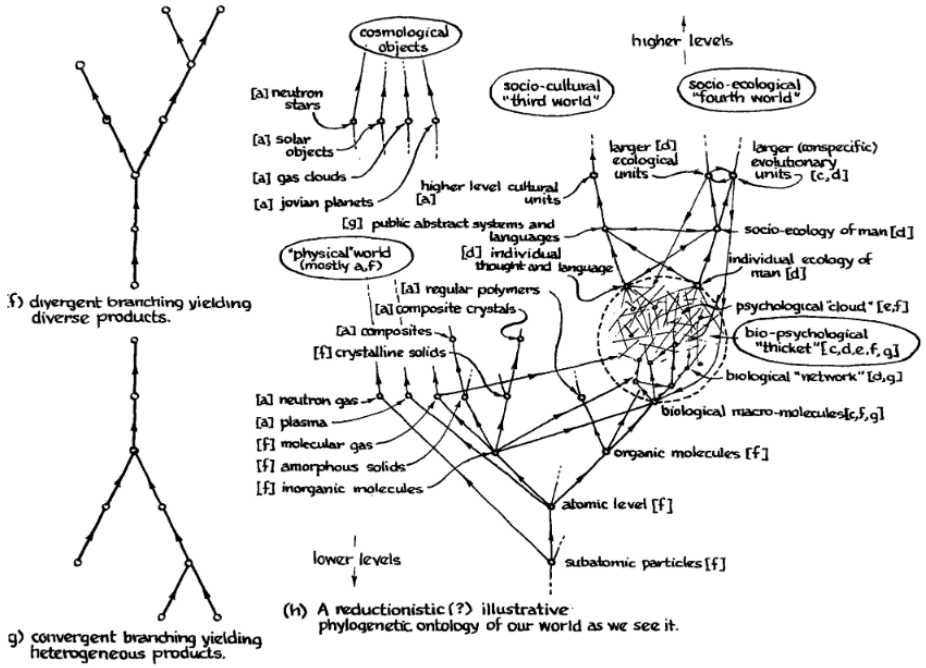



Figure 3.3: Causal Thickets

of those sciences are "causal thickets." In Figure 3.3, borrowed from [44], we can see Wimsatt's illustration of the complexity of a leveled conception of the sciences. There are a couple of features to note about this figure. First, notice that there are both convergent and divergent branching patterns as we move 'up' the levels. Thus, some higher-level[3] sciences (such as, inorganic and organic chemistry) diverge from lower-level sciences (such as, atomic physics); while some lower-level sciences (such as, what Wimsatt refers to as the "individual ecology of man" and "individual thought and language") converge into higher-level sciences (such as, what Wimsatt refers to as the "socio-ecology of man"). Second, notice that there are places where definite level demarcation is practically impossible. This is what Wimsatt refers to as "causal thickets," such as the "bio-psychological thicket."

I think it is by no means coincidental that the levels around the scale of reality at which humans reside is where we find the most complex causal thickets. Given the sort of conception of abstraction for which

[3]Keep in mind that all level talk is relative. In other words, a level is only considered to be a 'higher' level relative to some other 'lower' level.

I am advocating, I think this fact simply reflects our greater capacity for abstraction with respect to those 'things' that are similarly scaled with respect to ourselves. We have a clearer understanding of the objects of the biological sciences than the physical sciences at the extreme ends of the 'scale horizon' in which we find ourselves - namely, quantum physics and astronomy. From the perspective I'm developing, this is because the thicket of indefinitely demarcated levels is a mere symptom of the increased transformational capacity we have with respect to the various objects of the bio-psychological sciences. For example, we can more fluidly and more easily, and with more levels in between, move from thinking about the brain at the neuronal to the cognitive level than we can moving from the quantum-particle to the atomic level. This increased transformational capacity is a reflection of the fact that we have a better abstract understanding of the brain and what it does than an atom (or galaxy).

This is, of course, a controversial claim and defending it is beyond the scope of the thesis. The point here is merely suggestive. What I am really hoping to convey is that our capacity for transformational abstraction with respect to a specific domain can be more deeply understood as a capacity to transition from level to level. And the more sophisticated our understanding of the specific domain becomes, the more levels there are from and to which we can move. In this sense, I propose that transformational abstraction may be understood as a sort of 'level-hopping' within a 'representational thicket.' The circuits described above are essentially the structures within a network that account for its capacity to level hop.

There is a bit of a tension between the idealized conception of levels found in List's formal framework and the more subtle understanding of levels as causal thickets developed by Wimsatt in his analysis of the 'nitty-gritty' details of the biological sciences. Consequently, there is a tension in the work to which I have put the levels metaphor in the previous sections and the current section. From an idealized perspective, levels are, relatively speaking, definitely demarcated. Thus, the emergence of abstraction was conceptualized as the emergence of a higher level. From the representational thicket perspective, however, it is the *movement within a collection* of, relatively speaking, indefinitely demarcated levels that corresponds to an abstraction.

The key to resolving the tension is, again, scale. In short, what appears to be, at one scale, a constellation of (relatively speaking) indefinitely demarcated levels through which an intelligent agent is hopping may be understood as, at yet another scale, a (relatively speaking) definitely demarcated unitary level. Take, for example, Wimsatt's "bio-psychological thicket." At one scale, this a complex array of interdependent indefinitely distinct levels. At another scale, it's just one level in opposition to more fundamental levels, such as the atomic level.

This brings us back to the controversial point made above that we have a better abstract understanding of those processes at the relative scale at which we find ourselves. In short, I think there are complex causal thickets at both quantum and astronomical scales - we just don't have a very sophisticated understanding of them. Another way of stating this is that there aren't many levels to and from which to hop in our social understanding of those domains. In other words, the denser our representational thicket associated with some domain, the more we understand that domain and the more reliable our abstraction reasoning about with respect to that domain. This perspective should make it clear why the question of scale, along with the related issues concerning informational complexity and the parafinite, are crucial to explaining the success of neural net architectures like the transformer and specific models such as GPT-3.

# Chapter 4

# GPT-3 as semantic model

My attention in this chapter will be given to the inferentialist theory of meaning. The chapter concludes by contrasting models of the meaning of the logical connectives as described by Garson and connectionist models (such as GPT-3) of linguistic meaning. I argue that connectionist models of linguistic meaning are better suited to capture the indefinite nature of said meaning insofar as they distribute the meaning across circuits within the network. And as we saw in the previous chapter, these circuits allow for the network to realize a capacity for transformational abstraction. The discussion in this chapter will set the stage for a more detailed exploration into the indefinite nature of linguistic meaning in the next chapter.

## 4.1 Logical inferentialism

One of the central concerns in the philosophy of language is the question concerning meaning. In virtue of what do words and sentences get their meaning? Like most debates in philosophy, there are a number of contenders for *the* theory of meaning. And, like most debates in philosophy, there is little indication that any of the contending theories of meaning is likely to garner widespread support among philosophers, at least not any time soon. Instead, much of the discussions surrounding theories of meaning tend to operate within the context of a single theory (or, a small family of similar theories). Similarly, in the following, I will focus my attention solely on the inferentialist theory of meaning. This is not necessarily to advocate

for such a theory as *the* theory of meaning. Rather, my aim is simply to consider what we should expect an inferentialist theory of meaning to look like, specifically in light of recent developments in the modeling of natural language in AI (the most notable example, of course, being GPT-3). Although I will not be advocating for inferentialism as *the* theory of meaning, I do believe the relative success of GPT-3 qua semantic model lends credence to the inferentialist program in general.

Most of my attention will be given to a specific instance of inferentialism as presented by James Garson [17]. He develops a version of the narrower thesis known as logical inferentialism, which states that logical constants, in particular, get their meaning from the syntactical rules that govern the inferential roles they play in logic. Examples of logical constants include the logical connectives, modal operators, and so on. Garson is specifically concerned with the meaning of the logical connectives. He suggests that, if we hope to explain the meaning of natural language in terms of inferential roles and logical relations, we should, at least, be able explain the logical connectives in such terms. Thus, modeling the inferential role of the logical connectives provides a case study demonstrating what a more general inferentialist theory of meaning should look like.

Before discussing the details of Garson's project, I think it will be helpful to trace an outline of the critique I intend to develop. I agree with Garson that, if we believe inferential roles expressed in terms of logical relations are a sufficient model of the meaning of natural language, we should certainly be able to model the meaning of the logical connectives in those terms. But the success of a logical inferentialist program is not obviously a reason to be optimistic about the general inferentialist program. Even if logical relations alone are sufficient to determine the meaning of the logical connectives, we have no reason to believe that this acts as evidence that logical relations will be sufficient to determine extra-logical entities, such as words and sentences in natural language. Of course, the idea is that we can 'translate' such natural language entities into logical format, just as we have 'translated' the natural language correlates of the logical connectives into logical format. But, as I will argue, this begs the question – does such a model of the logical connectives capture the meaning of their natural language correlates? Or, does it simply capture the meaning of the connectives qua logical objects? I will argue in favor of the latter case.

So what does a logical inferentialist theory of meaning look like? For my purposes, Garson provides an excellent case study. But it should be noted that Garson's project is by no means standard. Whereas logical inferentialism is usually rooted in proof-theoretic approaches to semantics, Garson hopes to bridge a bit of the gap between the referentialist and inferentialist theories of meaning by developing a model-theoretic account of inferentialist semantics. In short, what he hopes to accomplish is a demonstration of exactly what truth-value conditions (which are typically considered the *reference* of propositions by referentialists) are expressed by the rules that govern the logical connectives. To be clear, however, Garson's goal is not to vindicate referentialism or to commit to the idea that the reference of a proposition is its truth-value. Rather, his goal is simply to take advantage of the conceptual tools (most notably, the concepts of model and truth) typically confined to the domains of (non-inferentialist) model theory and referentialist theories of meaning. In short, the inferentialist need not throw out the baby (in this case, conceptual notions such as model and truth) with the bath water (in this case, the unpalatable philosophical thesis of referentialism). The basic project is to examine what truth value conditions all valid models have in common; or in other words, what truth value conditions are 'expressed' by the system in question.

It turns out that the answer to this question depends on how we define model, what we take 'expression' to mean, and the framework within which we construct the rules. As Garson presents the debate, there are three choices for both of these categories, i.e., there are three conceptions of expression and three frameworks within which to construct the rules. Concerning expression, we have deductive, global, and local expression; and concerning rule formats, we have axiomatic, natural deduction, and multiple-conclusion sequent style. What Garson hopes to convince his readers is that the perfect balance is to be found by employing a natural deduction rule format in the context of global expression. What is unique about this combination is that the truth-value conditions turn out to be intuitionistic, a fact Garson seems to take as support for the view that intuitionistic logic is, at least in some sense, the *correct* logic. In contrast, Garson demonstrates that a multiple-conclusion sequent style rule format determines the classical truth conditions, regardless of how expression is defined. Likewise, he demonstrates that an axiomatic rule format underdetermines the truth conditions, again regardless of how expression is defined.

The arguments for the above conclusions, however, are beyond the scope of this paper. What is important is how, even in the case of truth-value underdetermination, the meaning of the logical connectives is rendered definite. In other words, none of these approaches, as I will argue below, allow for the meaning of the logical connectives to be indefinite. In fact, any logical approach to modeling the meaning of the logical connectives (or any linguistic entity for that matter) will render that meaning definite. Of course, if one thinks that the meaning of language is something that is definite, then this should not be considered a problem. Thus, the conclusion I hope to push the reader towards in this section is that the meaning is indefinite.

Before arguing for this thesis (i.e., that meaning is indefinite), I must first explain why and how any logical approach to semantic modeling will render meaning definite, even in cases where the meaning is underdetermined. Thus, the crucial distinction to be made here is that between underdetermination and indefiniteness. In short, just because the meaning of the logical connectives is underdetermined does not imply that their meaning is indefinite. I think the easiest way to demonstrate my point is to explain the underdetermination of the logical connectives in the context of deductive expression, and why this underdetermination does not equate to indefiniteness.

Let's start by defining deductive expression. As I mentioned earlier, Garson's approach is model-theoretic; therefore, the necessary jargon to be defined comes from model theory. The crucial concept of model theory is, as one unfamiliar with the theory might imagine, a model. In Garson's approach to model theory, a model simply consists of any set of valuations, where a valuation is a function from the set of well formed formulas to the Boolean set of truth values $\{t, f\}$. An argument is considered to be valid with respect to a model if that argument is satisfied by every valuation in that model. A model of some system is considered to be a deductive model of that system iff all of the provable arguments of that system are valid. And a system deductively expresses some property iff every deductive model of that system possesses that property. So the question with which we are concerned is - what property is it that all deductive models of some system (in our case, propositional logic) have in common? For example,

we might expect that all deductive models of propositional logic deductively express the classical truth conditions. As it turns out, however, they do not.

Consider, for example, the provability valuation, which is a valuation that, from a truth functional perspective, assigns $t$ to every provable proposition and $f$ to every other proposition. The classical truth conditions assign $t$ to the proposition $p \to q$ whenever $p$ is assigned the value $f$. If we assume $p$ and $q$ are distinct propositions, i.e. $p \neq q$, then it follows that $p \to q$ is not a provable assertion. But since neither $p$, $q$, nor $p \to q$ can be proven, the provability valuation will assign $f$ to each of these propositions. This, however, contradicts the fourth line of the classical truth conditions for the connective $\to$ wherein $p$ and $q$ are assigned $f$ and $p \to q$ is assigned $t$. Thus, there is a deductive model (namely, any model that includes the provability valuation) that does not satisfy the classical truth conditions for the fourth line of truth table for $\to$. And since there are other deductive models that express the classical truth conditions for the fourth line of the truth table, the truth conditions for this line are underdetermined. In contrast, the first three lines are fixed as every deductive model satisfies the classical truth conditions for those lines. Thus, the classical truth conditions for $\to$ are not determined in the context of deductive expression.

Some interesting properties to note about this project are that the truth conditions for the connectives $\&$ and $\bot$ are determined (and they are determined classically), while the truth conditions for $\neg$, $\vee$, $\to$, and $\leftrightarrow$ are underdetermined (i.e., within the context of deductive expression). They all share the property of being determined for the first three rows and being undetermined in the last row - negation, however which consists of only two rows in its standard truth table representation, is determined in the first row and underdetermined in the second row. However, if we fix the truth conditions for any one of these operators classically, then it turns out that the rest of the operators will also be fixed classically. This property is referred to as semantic holism [24].

## 4.2 The definite semantics of logical models

The rest of Garson's book works through the determination, or lack thereof, of the truth-value conditions in the context of different rule formats (primarily, natural deduction) and different conceptions of

expressivity (primarily, global expression). What they all have in common, as I intend to show below, is that the meaning of the logical connectives is rendered definite, even if it is left underdetermined.

Consider the case described above concerning the underdetermination of $\rightarrow$. If we choose to stick with the deductive conception of expression (which, to be clear, is not what Garson does), then what is that we are committing ourselves to? In short, we are committed to the meaning of the connective $\rightarrow$ underdetermining the truth-value reference of implicative propositions with both a false antecedent and consequent. What we are not committing ourselves to is an indefinite meaning of the connective $\rightarrow$. The meaning of $\rightarrow$ may be underdetermined, but the conditions in which it is underdetermined are definite, i.e., we know precisely when the truth-value reference of the implicative proposition has not been fixed.

Moreover, since we are constructing the meaning of these connectives within the context of a bivalent logic, we also know that, even if the precise truth-value reference of such an implicative proposition (i.e., an implication with both a false antecedent and consequent) is underdetermined, it must be either true or false in specific cases. Thus, the meaning of such implications is not indefinite. Rather, it simply depends on the specific valuation we are considering. In some cases, such implications will be true, while in others it will be false. But it will always be one or the other, and it will never be indeterminate within the context of a specific valuation. Moreover, it will always operate in this way, which is to say that it will only be underdetermined when both the antecedent and consequent are false; in all other scenarios, the truth conditions will be fixed.

This definiteness of meaning I am attributing to the connective $\rightarrow$ is simply a consequence of modeling the meaning of the connective logically. We could add a third truth-value (a project for which I am quite sympathetic) and designate that truth-value to be indeterminate, as is done within Kleene's 3-valued logic. But such an approach doesn't quite get us an indefinite conception of meaning. Rather, it simply introduces the idea that a truth-value assignment may be definitively indeterminate, a possibility that we don't get in bivalent logics. Even if we were to scrap discrete truth values and opt for a fuzzy logic, we must still rely on definite decision surfaces to model the relationship between atomics (e.g. $v(r) > 0.8$ iff $v(p) < 0.5$ and $v(q) > 0.5$ - or something like that).

For those that think that meaning is definite, the above argument is inconsequential. But if you share my view that the meaning of language is (at least to some extent) an indefinite phenomenon, then the above should lead you to question the viability of an inferentialist program that attempts to model linguistic meaning within a logical framework.

## 4.3 The indefinite semantics of connectionist models

And this brings us back to GPT-3, in particular, and connectionist models of natural language, in general. One of the most significant attributes of GPT-3 is the extent to which it tends to get syntax correct. From an inferentialist perspective, we may wonder whether or not GPT-3 can act as an inferential model of linguistic meaning. If inferential roles determine meaning, then we should expect GPT-3 to have, to some extent, captured the meaning of human language. I argue that, not only should we consider GPT-3 the best model of the meaning of natural language to date, we should also consider the failure of logic-based approaches and the success of connectionist approaches (like GPT-3) with respect to modeling natural language to be evidence in favor of the idea that linguistic meaning is indefinite.

The question I think the inferentialist should be asking is whether or not logic is even the right level of abstraction at which to model natural language. Garson does demonstrate that the meaning of the logical connectives is determined (and, if I am right above, rendered definite) by the syntactic rules that govern the connectives. But, to be sure, this doesn't really entail that the meaning of the English words 'and,' 'or,' and so on have been determined. Although, to be fair, it is certainly plausible that the English words 'and' and 'or' behave roughly equivalent to their logical counterparts. The conditional operators, however, lack single word, natural language counterparts that behave as reliably as 'and' and 'or,' and thus seem to be logical artifacts.

And this only begins the journey away from what Garson refers to as a "toy example" into the much higher dimensional and rough terrain underlying natural language semantics. In the next chapter, I provide a sketch of the indefinite nature of linguistic meaning. As I see it, this provides both a sort of justification for thinking of a complex model such as GPT-3 as a semantic model in the inferentialist spirit

and some insight regarding our epistemic relationship to such a model. First, however, I turn towards providing a formal background for thinking about connectionist networks as models that will highlight their capacity to realize indefinite representations in the next section.

## 4.4 Connectionist networks as models

In this section, I provide a brief sketch of a formal framework within which we can think of connectionist networks as extensions of the logical models described by inferentialists, such as Garson. It is suggested that models may be thought of as formal categories (defined below). Then, it is shown that, unlike simple logical models, connectionist models include higher-level functions that underlie their dynamic nature. Finally, it is speculated that this higher-level dynamism, in conjunction with the surpassing of parafinite thresholds, is what allows for transformational abstraction to emerge in connectionist models such as GPT-3.

Put simply, a model is a representation of some system that tells us something about what we should expect to be true about the system. Thus, generally speaking, models can be thought of as functions from one description of a system to another. Typically, we know the input description and we infer the output description assuming we have given at least some degree of credence to the model. As formulated by [17], a model is a set of valuations, where a valuation is a function from the set of well-formed formulas to a set of truth values. Thus, a model is a collection of functions, where each function moves from some formal description of a possible state of the system to a formal description (in the form of a truth-valuation) of whether that input state is a true description of the system. Consequently, a model formulated as such may be equivalently represented as a category, where a category is a collection of objects (e.g., sets) and arrows (or, functions).

More specifically, a category is a collection of objects and arrows, where all of the following properties hold:

- each arrow is associated with two objects, a domain and codomain

- (composition) for each arrow $f$ and $g$ such that the codomain of $f$ is the domain of $g$, there is some composite function $g \circ f$
- (identity) each object $A$ is associated with an identity arrow $1_A$
- (associativity) any two composite functions $h \circ (g \circ f)$ and $(h \circ g) \circ f$ are equivalent

For my purposes, these details aren't really important. For the simple logical model formulated by [17], composition and associativity are trivially satisfied insofar as all functions share the same domain and codomain and thus there are no composite functions. Identity can simply be stipulated without any significant consequence.

What is useful about the category-theoretic formulation for my purposes is that a connectionist network, learning algorithm and all, is also essentially a category. As shown in [14], a neural network is a monoidal category that maps functions within the category of parametrized functions to itself. This higher-level mapping is what is known as a functor, essentially a function that maps functions to other functions. In a neural network, this higher-level function (i.e., functor) is typically the backpropagation algorithm.

It is common to think of the model associated with a connectionist network as the final parametrized function found at the end of training that (ideally) best approximates the target function. But this conception of a connectionist model is static and is essentially reducible to a logical model as formulated by [17]. Moreover, this conception of a connectionist model is not consistent with what is most unique and productive about neural networks, namely their capacity to dynamically update their parameters (i.e., learn). And this is precisely what is captured by [14], wherein the entire learning algorithm, both the actual and all potential sequences of parameter updates, and the capacity to dynamically update is all included in a single formal entity, namely the monoidal category mapping (through backpropagation as a functor) a category of parametrized functions to itself.

Connectionist models, dynamically conceived, are better capable of internally representing the indefinite nature of meaning because, when sufficiently scaled, higher-level abstractions that are robust

to dynamic updates may emerge within network. It is precisely because these higher-level abstractions are robust to lower-level (i.e., parameter-level) changes that these higher-level abstractions achieve their indefinite status.

# Chapter 5

# The indefinite

There's a strong association between meaning and subjectivity. Although there have been attempts to strip consciousness and subjectivity away from linguistic meaning, there is nevertheless strong intuitive force behind the idea that a randomly generated novel by a more primitive primate thrashing away at a keyboard doesn't really mean anything. We tend to think that meaning must arise from an individual agent acting as the subject from which the meaning originates. Consequently, the suggestion that an artificially intelligent agent might actually mean something when it generates strings of symbols may seem preposterous upon first glance. Although the strings generated by GPT-3 seem meaningful on the surface, it is intuitively compelling (at least to me, and I'm pretty sure I'm not alone) that GPT-3 doesn't actually mean what it says. It's never experienced the world and all of the things in that world to which all of its generated symbols refer. In fact, as I argued in the previous chapter, GPT-3 seems to be better conceptualized as a model of meaning rather than a meaning-maker. In other words, meaning seems to come from us humans, and GPT-3 simply acts as a representation of what we mean – we're the meaning-makers, not our AI.

This, however, is only the beginning of the story, at least in my view. This is because it is committed to an inadequate conception of human agents as the originators of meaning. As I hope to demonstrate in this chapter, tracing the origin of what some string of symbols mean is much messier than simply identifying the agent from which the string originated. Within every utterance, there is an entire world, both unconscious and external to the agent, reflected in the meaning of the symbols shared. As Heidegger

[21] put it, "language speaks." In other words, language is a complex historical process that unfolds through us rather than some abstract depository from which we pull utterances based solely upon our own will and individual agency. This chapter takes a brief tour through a variety of philosophical sources arguing both in favor of an indefinite conception of linguistic meaning and an indefinite conception of the agential source of meaning, while giving particular emphasis to those sources of meaning that extend beyond the individual into the social and collective processes of which we are mere parts.

## 5.1 Tracing the source of meaning

I begin this journey into the origin (or non-origin, as we will shortly see) of meaning with a bit of Derridean thought. One of the difficulties facing any interpretation of Derrida is his use of 'master-words' that seem to be both the same and different. One of those such words is 'trace.' In an introduction to an English translation of Derrida's *Of Grammatology* [12], Spivak suggests that 'trace' is interchangeable with other Derridean concepts, such as 'arche-writing' and 'différance.' Thus, trace is to be understood as that which resides at the origin of writing (i.e., arche-writing) and should simultaneously be understood as that entanglement of differences that constitute the meaning of words (i.e., différance). Because this origin is an entanglement of differences, however, it is a non-origin. And this highlights the significance of Derrida's use of the word 'trace' - in short, the 'origin' of meaning is a mere trace, a vague impression of an underlying source that will never fully disclose the original. In fact, the idea that there is an original is mistaken - there are merely traces upon traces all the way down.

With this sort of philosophical framework in mind, we are in a better position to understand what an indefinite model of meaning should look like and why a definite (i.e., logical) approach to modeling meaning is unsatisfactory. By rendering the meaning of, for example, the logical connectives definite, the logical approach to modeling meaning forces us, in essence, to accept a narrative about the origin of those terms. As we saw in the discussion of Garson, this origin might be (if Garson has his way) grounded in intuitionistic logic. Or, if we prefer multiple-conclusion sequent rule format or a local conception of expression, this origin might be grounded in classical logic. These possibilities are not exhaustive, but they

all point to the same general structure of logical approaches to modeling meaning - namely, once we have decided upon the framework to be used, we are committed to a definitive origin.

The problem with this is that it conceals much more than it reveals. Perhaps the most obvious evidence of this is the fact that, depending upon the framework we choose, we are left with one conception of the meaning of the logical connectives in opposition to the others. But this points towards an important advantage of an indefinite approach - we need not commit ourselves to a single meaning of any given linguistic entity. And this is more advantageous, I think, because it is much more consistent with how we know natural language actually works. Words have complicated meanings that are often vague and shift over time. So if we are to look for the meaning of those words, we should expect to find a mere trace of an underlying entanglement of meanings (that is, something akin to a representational thicket).

## 5.2 Natural kinds

A similar sort of take on meaning can be found in Wittgenstein's *Philosophical Investigations* [45]. In perhaps one of the most famous passages of twentieth century philosophy, Wittgenstein suggests that the best way to characterize that which unites all 'games' is not a set of essential criteria but a collection of "family resemblances." Boundaries may be drawn. But they are typically vague and often aren't epistemically necessary for us to be able to competently use a term. This sort of conception of the meaning of a term is in contradistinction with the logical, i.e., set-theoretic, approach to defining a concept wherein precise criteria for the extension of some term are required.

Boyd [4] has argued that, in the context of biological categories, most notably the species category, necessary and sufficient criteria for membership simply cannot be provided. Instead, he suggests that what unites all members of a species together is a collection of what he refers to as "homeostatic property clusters." The basic idea is that what constitutes the meaning of the species concept is a cluster of properties that not only tend to be associated with that species but tend to causally reinforce each others' presence in that species. Few properties (if any) are necessary and rarely is any subset of properties collectively sufficient for species membership. In short, the species category is indefinite. Furthermore, the notion

of homeostatic property clusters corresponds directly with my conception of representational thickets. In fact, I see Boyd's argument that the species concept is a homeostatic property cluster as good reason to expect the abstraction of the species concept we hold collectively in our minds as a representational thicket.

Moreover, Boyd [5] extends the notion of homeostatic property clusters into the moral domain as well, claiming that "goodness" in humans is also a homeostatic property cluster. He argues that taking such a perspective on goodness vindicates moral realism insofar as it provides a naturalist understanding of the concept. Instead of some unreal ideal, goodness has an ontological status on par with concepts in the physical sciences (such as, species). For reasons that were intimated in the third chapter and for reasons beyond the scope of this thesis, I tend to think the ontological structure of all natural processes is akin to being a homeostatic property cluster. Thus, we should expect the abstract representation of those concepts to have a similar structure, which is what is intended in the notion of representational thickets.

## 5.3 Phenomenology

I think we can also find an argument for the indefinite nature of meaning in the phenomenological tradition, which traces its origins to the work of Husserl. Without diving into the rabbit hole that is Husserlian scholarship and interpretation, I think it is relatively uncontroversial to say that the key insight behind phenomenology is that experience (or, phenomena) should be investigated by analyzing the phenomenological structure that exists in the intentional relation between the subject and the objects of that subject's experience. In backlash against Kant's notion of noumena (or, at least, a popular understanding of Kant's noumena), phenomenology rejects the idea that there is a world out there independent of our experience (no matter how chaotic or uninterpretable). This rejection, however, isn't so much metaphysical or ontological as it is methodological (although that's not to deny the metaphysical and ontological conclusions phenomenologists will draw while employing the phenomenological method). The idea is that, what we are first and foremost is the certainty of our experiences (and here, we should be thinking of the certainty of the Cartesian cogito). All of our knowledge about the world must, at the end of the day, be understood

within this framework, for it is only through the phenomenal character of experience that we have access to the world. And this is what phenomenology seeks to do – that is, to provide a logic of sorts in which the phenomenal character of our interaction with the world can be understood. For Husserl, phenomenology was to act as a foundation for all of human knowledge.

Of course, much of this is an oversimplification and ignores the differences between the various phenomenologists. Nevertheless, I think there is a grain of truth in the above that is sufficient for our purposes. Heidegger develops an account of human being, or Dasein, that emphasizes the structural whole of being-in-the-world. We shouldn't think of human beings as isolated minds existing in an external world from which those minds remain distinct; rather, we should see that the world as it is revealed is fundamentally entangled in the being for whom being is revealed, namely us humans. This entanglement of the world, the objects in the world, and the beings for whom that world and being as a whole is revealed is something, according to Heidegger, that we must recognize and accept if we are to engage in ontological reflection. But since this entanglement with the world is dynamic, with old entanglements dying, new ones emerging, and the boundaries between different ways of engaging with the world constantly shifting, the various meanings that emerge from this worldly entanglement must be indefinitely constituted.

## 5.4 The Turing Test and the Chinese Room

Any discussion of what an artificially intelligent agent means would be incomplete without a discussion of the Turing test [42]. One of the most impressive features of GPT-3 is how close it comes to passing this test. In fact, I don't think it's entirely unreasonable to argue that GPT-3 has already passed the test, depending on how stringent we consider the criteria for passing the test to be. In any case, regardless of how high we set the bar, when we inquire into whether or not some artificially intelligent agent can pass the test, we take it for granted that the agent in question has clearly been identified. But the boundaries of an artificially intelligent agent are far from clear. Is the agent just the algorithm? Or, is the hardware part of the agent as well?

My intention is not to answer such questions, only to point out that answers to such questions must be assumed (whether explicitly or implicitly - usually implicitly) whenever we consider whether or not some agent has passed the Turing test. To demonstrate, consider the popular Chinese Room argument [37]. Imagine a person who doesn't know any Chinese in a room with access to a sufficiently large compendium of rules allowing them to transform strings of Chinese language input into the room into outputs that would reasonably convince a native Chinese speaker that the person in the room 'understands' Chinese. The basic argument is that this scenario is no different than a computer processor utilizing rules programmed by humans to transform input strings into output strings that would reasonably convince someone that a human must have produced the output strings. But just as the person inside of the Chinese room doesn't know Chinese, the computer doesn't know whatever language it is capable of producing.

But notice that this argument crucially relies on the analogy between the person inside of the room consulting the compendium of rules and the computer processor executing commands. One of the problems with this analogy is that I don't think anyone would claim that it is the computer processor that understands human language in the case of an artificially intelligent agent such as GPT-3. One plausible candidate is the model with all of its well trained parameters, which is more analogous to the compendium of rules than the person inside of the room. But we might also look for understanding, not in the person inside of the room, nor in the compendium of rules, but in the room itself.[1] It is incredibly difficult to identify the agential source of meaning. As we have seen above, there seems to be good reason to believe that humans are not always (and perhaps rarely - if ever) the agential sources of our own meaning. So why should we expect an algorithm, or a computer, or a robot, to be a clearly identifiable agential source of meaning?

[1] This is what is known as the "systems" reply to the Chinese Room argument. It is the response I favor. See [3, 25] for examples of this sort of response.

## 5.5 Agential sources

Perhaps the most comprehensive philosophical analysis of the history of the self can be found in Taylor's work, *Sources of the Self* [41]. Much of the book's historical breadth reflects the great pains to which Taylor goes in order to demonstrate that what we take the self to be is not how people in every culture throughout time and space have understood the self. For example, consider the basic idea that the self is internal, i.e., it is that which is within us, unavailable to other agents, which are external, i.e., out there in the world and not in here. Taylor remarks that this "localization" of the self as "inside" (in contrast to "outside") "is not a universal one, which human beings recognize as a matter of course… Rather it is a function of a historically limited mode of self-interpretation...." According to Taylor's narrative, the seed of this internalization of the self was sown by Plato, began to sprout in the work of Augustine, and reached fruition in the Cartesian *cogito*.[2]

We tend to assume that our own conception of the self, despite its lingering dualism and conceptual tensions, is the most natural. Jaynes [23], however, famously argued that, roughly speaking, pre-Homeric humans didn't even consider the entirety of their inner voice to be their own. Rather, those inner voices were attributed to the gods, which were distinct internal voices with distinct personalities. Jaynes argues that, initially, these voices would have been important leaders, but that, over time, as these leaders passed away, their voices would be culturally perpetuated in the minds of various people. Entire mythologies, religions, and world-views, he suggests, arose in this way.

What's really interesting, and most relevant for my purposes, about Jaynes' argument is that it implies that the ancient self was experienced quite differently than the way we experience our own selves. Jaynes argues that the mental framework of the characters in Homer's epics seems to suggest that they certainly did not identify as the voices (i.e., gods) in their head. In fact, he claims, when they would act in a way encouraged by a god, they would offer up the god's command as a sufficient justification for their behavior. In such situations (again, according to Jaynes), these justifications were taken at face value – responsibility

[2] See [41], Part II.

for an action commanded by the gods was never placed on the individual in the same way that we place responsibility on individual actors. In this respect, there's a tremendous amount of overlap in the work of Jaynes and Taylor – both reach the conclusion that the self is a culturally relative phenomena that is deeply entangled in the morality of that culture. [3]

In the contemporary world, we consider 'internal' voices that are experienced as outside of the boundaries of the self to be a symptom of an underlying disorder, most notably schizophrenia. Of course, as Hacking [20] has argued, the way in which schizophrenia is experienced today cannot be disentangled from the cultural interpretation of schizophrenia itself. Any sort of classification scheme concerned with human kinds, such as mental disorders, is subject to a looping effect wherein the way in which agents learn to classify themselves and their own social and psychological experiences will influence the way in which those categories persist and change over time. Jaynes argues that it is the very same neurological processes that allowed for early humans to hear the voices of gods that is operating under the surface in the case of schizophrenia. What's different is the way in which these voices are experienced and interpreted by both the individual and society at large.

Of course, this is certainly an oversimplification of Jaynes' argument. There's a lot going on in his work, and I can't possibly survey all of the nooks and crannies of his view. My point is simply that we have good reason to believe that our understanding of the self is not something that has been continuous across time. That's not to say that there isn't continuity with the past. Much of what Taylor is up to in his work is detailing the development of the modern self and tracing its genealogy back to some of its sources in the history of philosophy. But what we should take away from these thinkers is that the self is much more indefinite than we typically take it to be.

[3] Steiner [39] also argues that the experience of pre-Homeric man was quite different than our experience. There's an incredible amount of overlap in the historical arguments of Steiner, Jaynes, and Taylor. Fleshing out what these thinkers have in common would be an incredibly interesting and revealing analysis.

## 5.6 Buddhism, psychoanalysis, and the absence of self

Much stronger challenges to our conception of the self as a definitively individuated phenomenon arise from the Buddhist and psychoanalytic traditions. One of the central challenges of Buddhist philosophy is that the self and ego as they are typically experienced is illusory. Of course, Buddhist philosophy is quite diverse and anything beyond this proclamation would almost certainly lead to controversy among Buddhist philosophers. But, as a philosophical tradition with a history as rich and diverse as European philosophy, Buddhism offers an interesting case of a tradition that implicitly assumes a dogma (i.e., the dogma that there is no self) that stands in stark contrast to the dogma of the European philosophers (i.e., that there is such a thing as the self). All of this changes, however, when Sigmund Freud challenges the implicitly assumed conception of mind that permeates European intellectual history when he proposes the idea of the unconscious, i.e., that collection of processes that seems to be constantly operating 'within' us but outside of our awareness.

What both of these philosophical traditions have in common is an emphasis on how much of the processes that determine our thoughts and behaviors lies outside of our awareness, even if those processes are, in some sense, a part of us. Social and environmental processes have a tremendous influence over what we think and do. But the challenge from Buddhism and psychoanalysis is not that processes 'outside' of ourselves influence us, but that processes 'within' us, but of which we are not aware, are profoundly important for understanding how we think and what we do.

## 5.7 Wrapping up

What we have seen in this chapter is that the boundaries of the self that demarcate it from others and the world are far from clear. Much of what we attribute to our self seems to reflect our culturally relative conception of what it means to be a self. There are strong arguments that our linguistic utterances are spoken through us by cultural processes beyond our control. This sort of perspective can be found in Derrida, Heidegger, Jaynes, and Taylor, among others – this is just the sample of philosophers I have

chosen to review. Moreover, the results of phenomenology suggest that much of what is taken to be external to ourselves, i.e., that stuff out in the world, is actually deeply constitutive of who we are and what we take ourselves to be.

I think we can now see GPT-3 in new light. I argued that GPT-3 seems to satisfy much of the criteria we are looking for in an inferentialist model of meaning, with one exception – namely, the concepts of which GPT-3 is a model are not rendered definite in any way similar to how the logical constants are rendered definite for the logical inferentialist (even when the logical constants are indeterminate, this indeterminacy is rendered definite). But given the indefinite nature of the self, language, and the relationship between the two, I don't think we should be surprised that the best model of human linguistic meaning has an indefinite (or, inexact) representation of the concepts employed in human language. A similar point was made when I argued that understanding the ontological structure of natural processes as homeostatic property clusters lends credence to the notion that our abstract representations of those concepts are representational thickets.

Moreover, thinking of GPT-3 (and connectionist models in general) as a semantic inferentialist model fits well with the sort of picture that emerges from the conjunction of Derrida's conception of meaning as trace and Wittgenstein's conception of meaning as intrinsically entangled in the variety of games that we humans play. GPT-3 models our meaning, but it is far from identifying the trace (which is a necessarily impossible standard for a theory of meaning, at least according to a Derridean perspective). Likewise, it demonstrates competence at playing our language games, but it clearly doesn't understand how our language games are entangled in all of the other games we play. In other words, GPT-3 does just what we should want a model of linguistic meaning to do – it captures the meaning of our language without capturing the meaning of much else about human activity, which is to say that it isolates and identifies that which is unique about language among other human activities.

The problem, however, is that it's not clear that the model offers explanatory value. The definiteness of a logical inferentialist take on meaning serves us a neatly subdivided framework within which we can understand the meaning of our language. GPT-3 may help us generate meaning, but insofar as it fails to

simplify linguistic meaning, it cannot be thought of as a model fit to be considered a theory of meaning. Such a model should explain something to us. But how can a model with roughly 175 billion parameters really explain anything?

There a few remarks I'd like to make in response to this question. First, I think that GPT-3 and similar technologies should be understood, and likely integrated into the rest of our technological ecosystem, as a sort of indefinite programming language. The real significance of GPT-3 as a piece of technology is not the underlying technical details of how it works but what it can do. For example, GPT-3 can transform natural language descriptions into executable programming code. Progress on this front may soon provide us with a higher degree of control over our computers by allowing us to interact with them higher levels of abstraction. Second, perhaps we should simply interact with GPT-3 in order to reveal the meaning in models. In short, if we want to know what X means, let's just ask it - 'what does X mean?' – rather than trying to decode its internal representation of X. Finally, a method similar to DeepDream could be developed for extracting internal abstract representations of language from GPT-3.

# Bibliography


1. Alammar, J. The Illustrated Transformer [Blog post]. *Github.* `http://jalammar.github.io/illustrated-transformer/` (2018).
2. Bassler, O. B. *The Long Shadow of the Parafinite: Three Scenes from the Prehistory of a Concept* (Docent Press, 2015).
3. Boden, M. *Computer Models of the Mind* (Cambridge University Press, 1988).
4. Boyd, R. Homeostasis, species, and higher taxa. *Species: New Interdisciplinary Essays,* 141–185.
5. Boyd, R. How to Be a Moral Realist. *Essays on Moral Realism,* 181–228.
6. Brooks, R. Intelligence Without Representation. *Artificial Intelligence Journal* **47,** 139–159 (1991).
7. Brown, T. B. *et al.* Language Models are Few-Shot Learners. arXiv: `2005.14165 [cs.CL]` (2020).
8. Buckner, C. Empiricism without magic: Tranformational abstraction in deep convolutional neural networks. *Synthese* **12,** 1–34 (2018).
9. Buckner, C. & Garson, J. Connectionismn. *The Stanford Encyclopedia of Philosophy* (ed Zalta, E. N.) (2019).
10. Cicala, F. Perceptrons, Logical Functions, and the XOR Problem [Blog post]. *Towards Data Science.* `https://towardsdatascience.com/perceptrons-logical-functions-and-the-xor-problem-37ca5025790a` (2018).
11. Craver, C. Levels. *Open MIND* (eds Metzinger, T. & Windt, J. M.) (2015).
12. Derrida, J. *Of Grammatology* trans. by Spivak, G. C. (The John Hopkins University Press, 1976).

13. Fodor, J. Special Sciences (Or: The Disunity of Sciences as a Working Hypothesis). *Synthese* **28,** 87–115 (1974).

14. Fong, B., Spivak, D. I. & Tuyéras, R. Backprop as Functor: A compositional perspective on supervised learning. arXiv: 1711.10455 [math.CT] (2019).

15. Fukushima, K. Neocognitron: A self-organizing neural network model of pattern recognition unaffected by shift in position. *Cybernetics* **36,** 193–202 (1980).

16. Garcia-Morales, V. Quantum Mechanics and the Principle of Least Radix Economy. *Foundations of Phyics* **45,** 295–332 (2015).

17. Garson, J. *What Logics Mean* (Cambridge University Press, 2013).

18. Gauker, C. *Words and Images: An Essay on the Origion of Ideas* (Oxford University Press, 2011).

19. Graves, A., Wayne, G. & Danihelka, I. Neural Turing Machines. arXiv: 1410.5401 [cs.NE] (2014).

20. Hacking, I. The Looping Effects of Human Kinds. *Symposia of the Fyssen Foundation. Causal Cognition: A Multidisciplinary Debate,* 351–394 (1995).

21. Heidegger, M. The Way to Language. Trans. by Macquerrie, J. & Robinson, E. *Basic Writings* (ed Krell, D. F.) (1977).

22. Hoel, E. The Overfitted Brain: Dreams evolved to assist generalization. arXiv: 2007.09560 [q-bio.NC] (2020).

23. Jaynes, J. *The Origin of Consciousness in the Break Down of the Bicameral Mind* (Houghton Mifflin Company, 2000).

24. Jr., N. D. B. & Massey, G. J. Semantic Holism. *Studia Logica: An International Journal for Symbolic Logic* **49,** 67–82 (1990).

25. Kurzweil, R. *The Age of Spiritual Machines: When Computers Exceed Human Intelligence* (Penguin, 2000).

26. LeCun, Y. *The MNIST Database* http://yann.lecun.com/exdb/mnist/.

27. List, C. Levels: Descriptive, Explanatory, and Ontological. *Nous* **53** (2018).

28. List, C. *Why Free Will is Real* (Harvard University Press, 2019).

29. List, C. & Pivato, M. Emergent Chance. *Philosophical Review* **124,** 119–152 (1 2015).

30. Locke, J. *An Essay Concerning Human Understanding* (The Pennsylvania State University, 1690).

31. McCulloh, W. S. & Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5,** 115–137.

32. Minsky, M. & Papert, S. *Perceptrons: An Introduction to Computational Geometry* (The MIT Press, 1969).

33. Olah, C. *et al.* Zoom In: An Introduction to Circuits. *Distill.* https://distill.pub/2020/circuits/zoom-in (2020).

34. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. Learning representations by back-propagating errors. *Nature* **323,** 533–536 (1986).

35. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (eds Rumelhart, D. E. & McClelland, J. L.) chap. 8 (Cambridge MIT Press, 2004).

36. Sabour, S., Frosst, N. & Hinton, G. Dynamic Routing Between Capsules. arXiv: 1710.09829 (2017).

37. Searle, J. Minds, Brains, and Programs. *Behavioral and Brain Sciences* **3,** 417–457.

38. Sharma, S. A Visual Introduction to Neural Networks [Blog post]. *Towards Data Science.* https://towardsdatascience.com/a-visual-introduction-to-neural-networks-68586b0b733b (2017).

39. Steiner, R. *The Riddles of Philosophy* trans. by Koelln, F. C. A. (Anthroposophic Press, 1973).

40. Szegedy, C. *et al.* Going deeper with convolutions. arXiv: 1409.4842 [cs.CV] (2014).

41. Taylor, C. *Sources of the Self: The Making of the Modern Identity* (Harvard University Press, 1989).

42. Turing, A. Computing Machinery and Intelligence. *Mind* **LIX,** 433–460 (236 1950).

43. Vaswani, A. *et al.* Attention Is All You Need. *Advances in Neural Information Processing Systems.* `http://arxiv.org/abs/1706.03762` (2017).

44. Wimsatt, W. C. The Ontology of complex systems: Organization, perspectives, and causal thickets. *Canadian Journal of Philosophy* **24** (1994).

45. Wittgenstein, L. *Philosophical Investigations* trans. by Anscombe, G. E. M. (Basil Blackwell, 1958).