Received: 12 December 2018 | Revised: 24 April 2019 | Accepted: 28 June 2019
DOI: 10.1111/phc3.12625

ARTICLE

WILEY

# Deep learning: A philosophical introduction

**Cameron Buckner**

The University of Houston

**Correspondence**
Cameron Buckner, Department of Philosophy,
University of Houston, Agnes Arnold Hall,
3553 Cullen Blvd., Room 513 Houston, TX
77204-3004.
Email: cjbuckner@uh.edu

## Abstract

Deep learning is currently the most prominent and widely successful method in artificial intelligence. Despite having played an active role in earlier artificial intelligence and neural network research, philosophers have been largely silent on this technology so far. This is remarkable, given that deep learning neural networks have blown past predicted upper limits on artificial intelligence performance—recognizing complex objects in natural photographs and defeating world champions in strategy games as complex as Go and chess—yet there remains no universally accepted explanation as to why they work so well. This article provides an introduction to these networks as well as an opinionated guidebook on the philosophical significance of their structure and achievements. It argues that deep learning neural networks differ importantly in their structure and mathematical properties from the shallower neural networks that were the subject of so much philosophical reflection in the 1980s and 1990s. The article then explores several different explanations for their success and ends by proposing three areas of inquiry that would benefit from future engagement by philosophers of mind and science.

## 1 | INTRODUCTION: A CALL TO ACTION

Deep learning is currently the most powerful and high-profile approach to artificial intelligence (AI). Major technology companies like Google, Microsoft, Facebook, and Amazon have all devoted marquee research and development groups to the study of the technology; Google alone reported having over 1,000 deep learning projects in the works as of 2016. Today, such systems already control operations as diverse as labeling images, recognizing speech, translating texts, playing strategy games, predicting protein folds, detecting new exoplanets, analyzing fMRI data, and driving automobiles autonomously. These feats clear a variety of benchmarks that skeptics supposed would remain beyond the reach of AI: They can identify complex objects in natural photographs at human-level performance, and they have defeated human champions in games as complex as chess, Go, and Starcraft II (Silver et al., 2018; Vinyals et al., 2019).[1] In addition to their practical achievements, deep learning networks are currently regarded as the best models of perceptual similarity judgments in primates (Yamins & DiCarlo, 2016).[2] Influential figures in deep learning research—whose achievements are featured monthly in prestige journals like *Science* and *Nature* and have been covered heavily in the scientific press since at least 2010—have offered ambitious claims about the philosophical and social significance of these achievements, such as that they vindicate classical empiricism about reasoning or that vast swathes of the human labor force will soon be rendered obsolete (Makridakis, 2017; Silver et al., 2017). Yet there remains significant debate in the technical literature as to why these networks perform so well or how their processing relates to human and animal intelligence.

It is therefore surprising that, as of 2018, there has been little discussion of the details of deep learning in philosophy journals that would distinguish it from other methods in machine learning and artificial intelligence.[3] The omission has been especially noticeable in philosophy of mind and cognitive science, which once engaged so extensively with shallow neural networks in the 1980s and 1990s. A commonly encountered attitude in these areas is that deep neural networks are just "more of the same"—perhaps an important engineering advance, but incremental rather than game changing—and so recent research developments do not merit the kind of careful scrutiny from philosophers that earlier waves of connectionism received.

In this article, I aim to temper this attitude by explaining the distinctive computational power of deep neural networks. I begin Section 2 by summarizing the main features that distinguish the most common and reliably successful deep architecture, deep convolutional neural networks (hereafter DCNNs), from their shallower forebears. In Section 3, I consider explanations for the distinctive success of these networks. In Section 4, I provide a list of questions for future philosophical research into the significance and philosophical implications of deep learning for philosophy of mind and science. Each section aims to briefly introduce the issues in a way that encourages reflection and participation by philosophers in future research.

## 2 | DCNNS: MAIN FEATURES

I now illustrate the main features of the most common and reliably successful deep architecture, DCNNs, by contrasting them with their forebears from the "Golden Age" of neural network research from 1980 to 1995 (in the interests of space, we will completely forego history, including stock criticisms of earlier neural architectures, adequately covered elsewhere—Buckner & Garson, 2018; Schmidhuber, 2015). All neural networks can be thought of as composed of nodes and links, intended to model the behavior of neurons and synapses at some level of abstraction. Processing is performed by passing an input signal to some array of input nodes, which then compute their assigned activation functions and pass an output signal up to the next layer in the network along their links, modified by those links' "weights" (which might produce inhibition, should the resulting value be negative). Activation propagates forward in this manner until it reaches a designated output layer of nodes, which is then decoded and taken as the network's "decision" on that input.

An exciting feature of these networks is their ability to discover novel solutions directly from problem data. The most popular learning algorithm since the 1980s has been error backpropagation learning. This method is called "supervised," because it deploys a teaching signal generated by an error function to calculate the distance between the actual and desired output for some examples, determined by a training set labeled with the correct answers. For multilayer networks, that error signal is then further backpropagated through the next previous layer of the network and used to adjust its input link weights, and so on until the error signal reaches the initial layer. Simply by backpropagating error signals and gradually adjusting link weights in this manner, network performance can converge on the solutions to a wide range of classification and decision problems.

These basic features are shared by most Golden Age and state-of-the-art DCNN models. Now for the contrasts. Let us characterize the typical Golden Age network by three properties; they are (a) *shallow*, with no more than three or four layers between input and output; (b) *uniform*, with only one type of node deploying a sigmoidal activation function; and (c) *fully connected*, with each node from a lower layer connected to each other node in the next layer up. The typical state-of-the-art DCNN, on the other hand, is (a) *deep*, containing anywhere from 5 to 250 (or more) layers; (b) *heterogeneous*, containing different kinds of processing nodes deploying different activation functions (especially convolutional nodes, rectified linear units or “ReLUs,” and nonlinear downsamplers like max poolers—each of which will be explained below); (c) *sparsely connected*, with later layers only taking input from nearby nodes with overlapping spatial receptivity from the previous layer (a fully-connected output layer is the common exception); and (d) deploys *explicit techniques to avoid overfitting* (such as dropout, which will also be explained below). Each of these features likely plays a role in making DCNNs orders of magnitude more efficient than shallower networks on the kinds of problems on which they reliably succeed.

## 2.1 | Depth

Neural networks can be understood as devices that efficiently compose complex categorization and decision functions from interconnections among simpler ones (i.e., the activation functions of their nodes). The most obvious distinguishing mark of DCNNs, qua members of the set of network-based, function-building devices, is their depth. Deepening networks was previously shown to have profound computational benefits; in the 1980s, the transition from two-layer perceptrons to three-layer networks with a “hidden layer” allowed neural networks to compute linearly inseparable functions such as XOR that had been shown to be beyond the reach of two-layer perceptrons (Minsky & Papert, 1969). Though the advantage of further depth is not quite so obvious as allowing us to compute an entirely new class of function, it has other benefits that we are only now beginning to understand.

Before discussing these additional benefits from a technical perspective, it may help to begin with an analogy. Consider a discussion between two technologists regarding the discovery of assembly line mass production of automobiles, circa 1920. Suppose one of the technologists is skeptical of the assembly line's significance, noting that any automobile that can be made by an assembly line could in principle be constructed by a team of skilled machinists with the appropriate plans, manually building one part at a time. “There is nothing so profound here,” the skeptic might say to his colleague, “as the advance from the steam engine to the internal combustion engine; at best, there could be mere efficiency gains.” The enthusiastic technologist, on the other hand, will note that with the same time and effort that it takes a team of skilled machinists to produce one automobile, an assembly line could produce tens of thousands. Moreover, once factories became more specialized, and a distributed, hierarchical supply chain and distribution network were established, the same small, intricate parts could be repeatedly reused in many different components, and in many different models of cars. After each worker grew increasingly specialized at doing a small range of simpler tasks reliably and efficiently, the whole network of factories could produce millions of automobiles in dozens of different models in the same period of time. Moreover, automobiles assembled in this way will be very different in their internal construction from automobiles which were designed to be assembled manually; they can be much more complex, containing many smaller and more specialized parts, which would become standardized and interchangeable at many levels of composition. It will also become much easier to design new automobile models in the future by reassembling those interchangeable parts in new ways. Suddenly, a highly-complex, fast, reliable, efficient automobile could be made available to every household—instead of just a few primitive, bespoke curiosities in the hands of privileged collectors and inventors. These advances are what allowed automobiles to change human society, and the enthusiast will regard their introduction as a breakthrough achievement.

Similar gains in the efficiency and complexity of representational schemes and decision-making policies are afforded by additional depth in neural networks. Specifically, deeper networks can solve certain types of classification and decision problems exponentially more efficiently than shallower networks. To demonstrate this possibility with more precision, I will first explain a simple class of deep network where exponential benefits in computational efficiency have been formally proven: sum-product networks (Delalleau & Bengio, 2011). We will then approach the question—which is to some degree still an empirical conjecture, explored at more length in Section 3—about the similarity such results bear to the more complex DCNN architectures operating on the kinds of perceptual classification problems on which they reliably succeed.

A sum-product network is a simple device for computing polynomial functions. These networks contain only two types of nodes, which return either a weighted sum or a product of its inputs. Now, let us compare the ability of a shallow network—with only one sum and product layer, respectively—to solve the same polynomial function as a deep network (Figure 1). A deeper architecture can more efficiently compose many different products of its input variables; specifically, the number of times a product composed at an earlier layer can be reused in more complex products built by later layers increases exponentially with the network's depth. This proposition has been proven for several special cases of sum-product networks, and the proofs all exploit the fact that the deeper sum-product network efficiently represents and computes a *factorized* expression of the polynomial function, whereas the shallow networks must compute an *expanded* expressions of that function—like the skilled but inefficient machinists—one product at a time. Thus, functions that permit such factorization—and perhaps more generally, functions that can be efficiently represented as redeploying simpler computations to hierarchically compose more complex computations—can be represented and computed exponentially more efficiently in a deep architecture than a shallow one.

## 2.2 | Heterogeneity

While depth brings similar benefits to state-of-the-art DCNNs as it does to sum-product networks, the former are built from nodes that each compute more elaborate functions than products and sums: namely, convolution, rectification, and max pooling (Figure 2). These three kinds of nodes are often layered in a series, with a convolutional node passing input to a rectification node (or "ReLUs," for rectified linear unit), and then a max-pooler taking input from several convolutional/ReLU combinations with overlapping spatial or temporal sensitivity. Understanding the cooperation between these different activation functions in series is crucial to understanding the distinctive power of DCNNs, so I elaborate each in turn (for a longer discussion, see Buckner, 2018). Doing so will also, to extend the assembly line analogy, help us understand what it is that DCNNs might "build."

Convolution is a linear algebra operation that transforms some chunk of the network's input to amplify certain values and minimize others.[4] In DCNNs, it is typically applied to a "window" of perceptual input data, such as a rectangle of pixels in an image or a snippet of audio information in a sound file (for ease of exposition, I will limit discussion to images in what follows). Pixels are themselves usually vectors of RGB color channel information at that point; the convolution operation returns a transformed matrix of RGB vectors that amplifies the presence of a certain feature, such as (at early layers) contrasts or shadings. These convolutional units are called "filters" or "kernels" (Figure 3). The output of each convolution operation is then typically passed to a ReLU unit, which activates according to a simple function called rectification if the result of the convolution exceeds a certain threshold. This is sometimes called the "detector" stage of processing, as activation is only passed up the hierarchy if the kernel's activation indicates that it found that feature at that location. To illustrate by example, suppose we had a kernel that amplified vertical lines; if the "window" of this convolution operation were tiled over the whole image and each passed to a different ReLU node, one chunk at a time, it would "filter" the raw input and return a transformed image that showed all and only the image's vertical lines.

Much of the distinctive power of DCNNs, however, is to be found in their ability to detect features in a variety of different locations and poses. In other words, we typically do not just want to detect vertical lines in a specific location, but rather lines in any orientation in any location. This can be achieved by passing each filter + ReLU output to a third kind of node whose function is to aggregate and downsample the activity of several different filters with overlapping spatial receptivity.[5] The most popular downsampling function in state-of-the-art DCNNs is max pooling, which sends up activation only from its most highly-activated input (Figure 4). In other words, if a max pooling unit receives input from a vertical line kernel and a horizontal line kernel at a particular location, then it will only pass

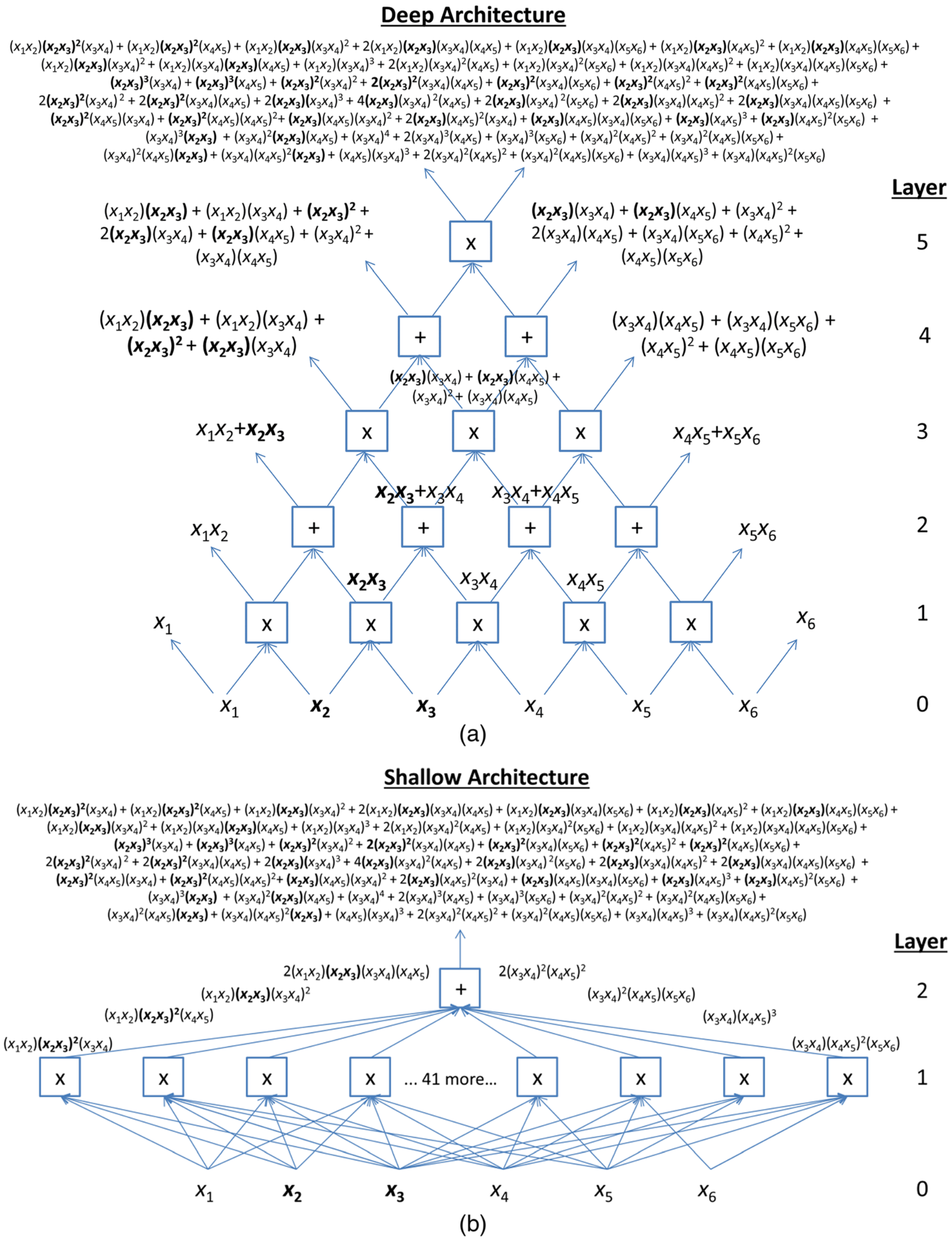

Deep Architecture
$(x_1x_2)(x_2x_3)^2(x_3x_4) + (x_1x_2)(x_2x_3)^2(x_4x_5) + (x_1x_2)(x_2x_3)(x_3x_4)^2 + 2(x_1x_2)(x_2x_3)(x_3x_4)(x_4x_5) + (x_1x_2)(x_2x_3)(x_3x_4)(x_5x_6) + (x_1x_2)(x_2x_3)(x_4x_5)^2 + (x_1x_2)(x_2x_3)(x_4x_5)(x_5x_6) +$
$(x_1x_2)(x_2x_3)(x_3x_4)^2 + (x_1x_2)(x_3x_4)(x_2x_3)(x_4x_5) + (x_1x_2)(x_3x_4)^3 + 2(x_1x_2)(x_3x_4)^2(x_4x_5) + (x_1x_2)(x_3x_4)^2(x_5x_6) + (x_1x_2)(x_3x_4)(x_4x_5)^2 + (x_1x_2)(x_3x_4)(x_4x_5)(x_5x_6) +$
$(x_2x_3)^3(x_3x_4) + (x_2x_3)^3(x_4x_5) + (x_2x_3)^2(x_3x_4)^2 + 2(x_2x_3)^2(x_3x_4)(x_4x_5) + (x_2x_3)^2(x_3x_4)(x_5x_6) + (x_2x_3)^2(x_4x_5)^2 + (x_2x_3)^2(x_4x_5)(x_5x_6) +$
$2(x_2x_3)^2(x_3x_4)^2 + 2(x_2x_3)^2(x_3x_4)(x_4x_5) + 2(x_2x_3)(x_3x_4)^3 + 4(x_2x_3)(x_3x_4)^2(x_4x_5) + 2(x_2x_3)(x_3x_4)^2(x_5x_6) + 2(x_2x_3)(x_3x_4)(x_4x_5)^2 + 2(x_2x_3)(x_3x_4)(x_4x_5)(x_5x_6) +$
$(x_2x_3)^2(x_4x_5)(x_3x_4) + (x_2x_3)^2(x_4x_5)(x_4x_5)^2 + (x_2x_3)(x_4x_5)(x_3x_4)^2 + 2(x_2x_3)(x_4x_5)^2(x_3x_4) + (x_2x_3)(x_4x_5)(x_3x_4)(x_5x_6) + (x_2x_3)(x_4x_5)^3 + (x_2x_3)(x_4x_5)^2(x_5x_6) +$
$(x_3x_4)^3(x_2x_3) + (x_3x_4)^2(x_2x_3)(x_4x_5) + (x_3x_4)^4 + 2(x_3x_4)^3(x_4x_5) + (x_3x_4)^3(x_5x_6) + (x_3x_4)^2(x_4x_5)^2 + (x_3x_4)^2(x_4x_5)(x_5x_6) +$
$(x_3x_4)^2(x_4x_5)(x_2x_3) + (x_3x_4)(x_4x_5)^2(x_2x_3) + (x_4x_5)(x_3x_4)^3 + 2(x_3x_4)^2(x_4x_5)^2 + (x_3x_4)^2(x_4x_5)(x_5x_6) + (x_3x_4)(x_4x_5)^3 + (x_3x_4)(x_4x_5)^2(x_5x_6)$
Layer
$(x_1x_2)(x_2x_3) + (x_1x_2)(x_3x_4) + (x_2x_3)^2 + 2(x_2x_3)(x_3x_4) + (x_2x_3)(x_4x_5) + (x_3x_4)^2 + (x_3x_4)(x_4x_5)$
x
$(x_2x_3)(x_3x_4) + (x_2x_3)(x_4x_5) + (x_3x_4)^2 + 2(x_3x_4)(x_4x_5) + (x_3x_4)(x_5x_6) + (x_4x_5)^2 + (x_4x_5)(x_5x_6)$
5
$(x_1x_2)(x_2x_3) + (x_1x_2)(x_3x_4) + (x_2x_3)^2 + (x_2x_3)(x_3x_4)$
+
+
$(x_3x_4)(x_4x_5) + (x_3x_4)(x_5x_6) + (x_4x_5)^2 + (x_4x_5)(x_5x_6)$
4
$(x_2x_3)(x_3x_4) + (x_2x_3)(x_4x_5) + (x_3x_4)^2 + (x_3x_4)(x_4x_5)$
$x_1x_2+x_2x_3$
x
x
x
$x_4x_5+x_5x_6$
3
$x_2x_3+x_3x_4$
$x_3x_4+x_4x_5$
$x_1x_2$
+
+
+
+
$x_5x_6$
2
$x_2x_3$
$x_3x_4$
$x_4x_5$
$x_1$
x
x
x
x
x
$x_6$
1
$x_1$
$x_2$
$x_3$
$x_4$
$x_5$
$x_6$
0
(a)
Shallow Architecture
$(x_1x_2)(x_2x_3)^2(x_3x_4) + (x_1x_2)(x_2x_3)^2(x_4x_5) + (x_1x_2)(x_2x_3)(x_3x_4)^2 + 2(x_1x_2)(x_2x_3)(x_3x_4)(x_4x_5) + (x_1x_2)(x_2x_3)(x_3x_4)(x_5x_6) + (x_1x_2)(x_2x_3)(x_4x_5)^2 + (x_1x_2)(x_2x_3)(x_4x_5)(x_5x_6) +$
$(x_1x_2)(x_2x_3)(x_3x_4)^2 + (x_1x_2)(x_3x_4)(x_2x_3)(x_4x_5) + (x_1x_2)(x_3x_4)^3 + 2(x_1x_2)(x_3x_4)^2(x_4x_5) + (x_1x_2)(x_3x_4)^2(x_5x_6) + (x_1x_2)(x_3x_4)(x_4x_5)^2 + (x_1x_2)(x_3x_4)(x_4x_5)(x_5x_6) +$
$(x_2x_3)^3(x_3x_4) + (x_2x_3)^3(x_4x_5) + (x_2x_3)^2(x_3x_4)^2 + 2(x_2x_3)^2(x_3x_4)(x_4x_5) + (x_2x_3)^2(x_3x_4)(x_5x_6) + (x_2x_3)^2(x_4x_5)^2 + (x_2x_3)^2(x_4x_5)(x_5x_6) +$
$2(x_2x_3)^2(x_3x_4)^2 + 2(x_2x_3)^2(x_3x_4)(x_4x_5) + 2(x_2x_3)(x_3x_4)^3 + 4(x_2x_3)(x_3x_4)^2(x_4x_5) + 2(x_2x_3)(x_3x_4)^2(x_5x_6) + 2(x_2x_3)(x_3x_4)(x_4x_5)^2 + 2(x_2x_3)(x_3x_4)(x_4x_5)(x_5x_6) +$
$(x_2x_3)^2(x_4x_5)(x_3x_4) + (x_2x_3)^2(x_4x_5)(x_4x_5)^2 + (x_2x_3)(x_4x_5)(x_3x_4)^2 + 2(x_2x_3)(x_4x_5)^2(x_3x_4) + (x_2x_3)(x_4x_5)(x_3x_4)(x_5x_6) + (x_2x_3)(x_4x_5)^3 + (x_2x_3)(x_4x_5)^2(x_5x_6) +$
$(x_3x_4)^3(x_2x_3) + (x_3x_4)^2(x_2x_3)(x_4x_5) + (x_3x_4)^4 + 2(x_3x_4)^3(x_4x_5) + (x_3x_4)^3(x_5x_6) + (x_3x_4)^2(x_4x_5)^2 + (x_3x_4)^2(x_4x_5)(x_5x_6) +$
$(x_3x_4)^2(x_4x_5)(x_2x_3) + (x_3x_4)(x_4x_5)^2(x_2x_3) + (x_4x_5)(x_3x_4)^3 + 2(x_3x_4)^2(x_4x_5)^2 + (x_3x_4)^2(x_4x_5)(x_5x_6) + (x_3x_4)(x_4x_5)^3 + (x_3x_4)(x_4x_5)^2(x_5x_6)$
Layer
$2(x_1x_2)(x_2x_3)(x_3x_4)(x_4x_5)$
+
$2(x_3x_4)^2(x_4x_5)^2$
$(x_1x_2)(x_2x_3)(x_3x_4)^2$
$(x_3x_4)^2(x_4x_5)(x_5x_6)$
2
$(x_1x_2)(x_2x_3)^2(x_4x_5)$
$(x_3x_4)(x_4x_5)^3$
$(x_1x_2)(x_2x_3)^2(x_3x_4)$
$(x_3x_4)(x_4x_5)^2(x_5x_6)$
x
x
x
x
... 41 more...
x
x
x
x
1
$x_1$
$x_2$
$x_3$
$x_4$
$x_5$
$x_6$
0
(b)

**FIGURE 1** While a shallow network can compute the same function as a deep network, it can often do so only with exponentially more nodes (and derivatively, computations). These figures depict a simple sum-product architecture (extending an example from Bengio, 2009). The exponential gains in efficiency afforded by depth (with results formally proven in Delalleau & Bengio, 2011) derive from the fact that a deep architecture can represent a factorized expression of a complex polynomial (in this case, $(((x_1x_2 + x_2x_3))*((x_2x_3) + (x_3x_4)) + ((x_2x_3 + x_3x_4)*(x_3x_4 + x_4x_5)))*(((x_2x_3 + x_3x_4)*(x_3x_4 + x_4x_5)) + ((x_3x_4 + x_4x_5)*(x_4x_5 + x_5x_6)))$, whereas a shallow architecture can only represent its expanded expression. Since the number of unique products in the expanded expression grows exponentially with the number of factors, the number of nodes required by a shallow net to compute the same function grows exponentially with its deep counterpart's depth. Relatedly, a single computation performed at an earlier layer can be deployed exponentially often in later layers, relative to the network's depth (in the figure, the $x_2x_3$ product computed at Layer 1 has been bolded to illustrate this)

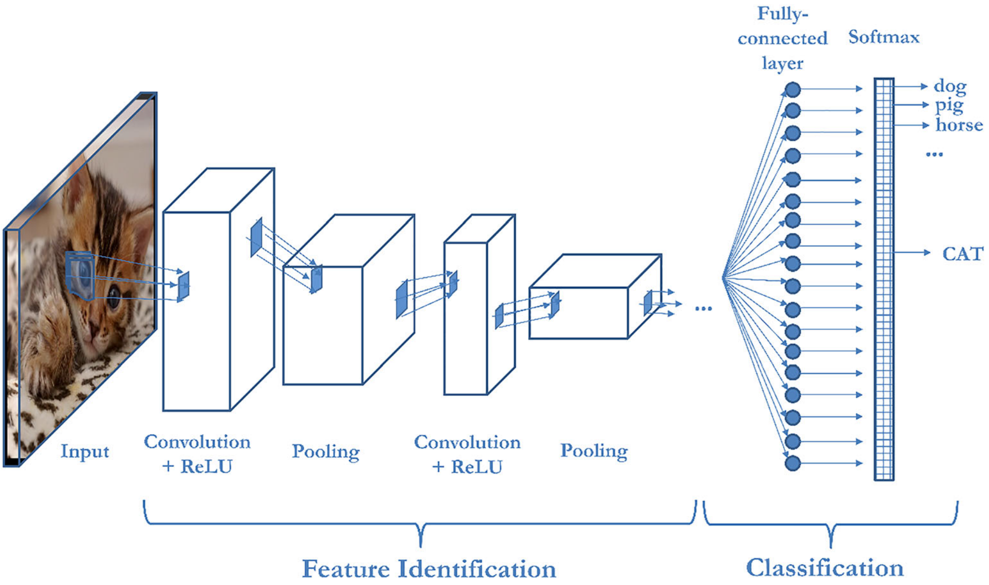


**FIGURE 2** The generic processing flow of a deep convolutional neural network, whose architecture involves transforming input signals through many sequences of locally-connected convolutional, ReLU, and pooling nodes, before finally passing them to a fully connected classification layer that assigns final category labels

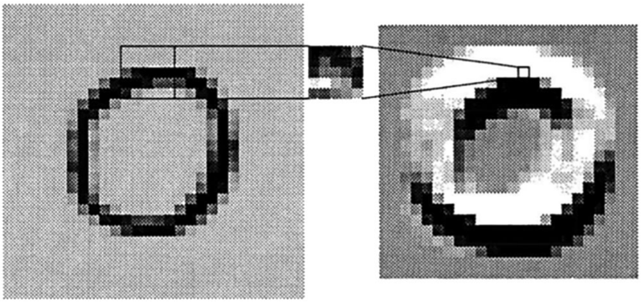
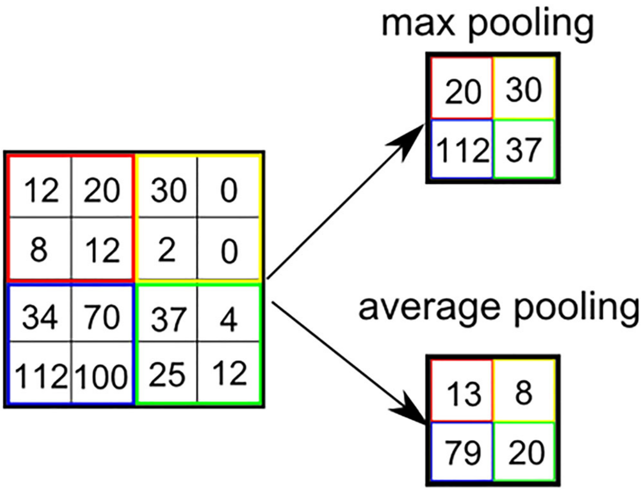


**FIGURE 3** An example of the visual output of a single convolutional kernel on real handwritten digit data in a trained convolutional network (reproduced from LeCun et al., 1990, 399). This (learned) kernel (in center) detects something like curves at the top or bottom of a digit (with source data on the left and transformed output on the right—often called a "feature map")



| | | | |
|---|---|---|---|
| 12 | 20 | 30 | 0 |
| 8 | 12 | 2 | 0 |
| 34 | 70 | 37 | 4 |
| 112 | 100 | 25 | 12 |

max pooling

| | |
|---|---|
| 20 | 30 |
| 112 | 37 |

average pooling

| | |
|---|---|
| 13 | 8 |
| 79 | 20 |

**FIGURE 4** A comparison of max pooling with average pooling for downsampling across activation received from the same receptive fields (reproduced from Singhal, 2017)

activation to the next layer for whichever of the two was most strongly activated (i.e., it forces the layer to make a "decision" as to which feature was most likely to be found at that location). Combining all three operations, we could produce a simplified, transformed representation of the source image which consisted not only of vertical lines in a particular spot but also of all lines in the original image in any location or orientation (Figure 5). These abstracted features then become available for processing at the next layer of the network, which performs a similar series of operations to detect yet more complex features. For example, the next layer of convolutional units could then build filters to detect angles from the transformed and simplified information about lines and their locations. This task would be much easier, from a computational perspective, than trying to build an angle detector from raw pixel information directly.

## 2.3 | Sparse connectivity

A third characteristic feature of DCNNs is their sparse connectivity between layers. Whereas nodes in Golden Age networks were often fully connected to each node in the layers above and below, nodes in DCNNs are usually only locally connected to nodes with spatially or temporally overlapping input receptivities, as if retinotopically (the only exception is a fully-connected layer that typically precedes the final categorization decision). This sparse connectivity

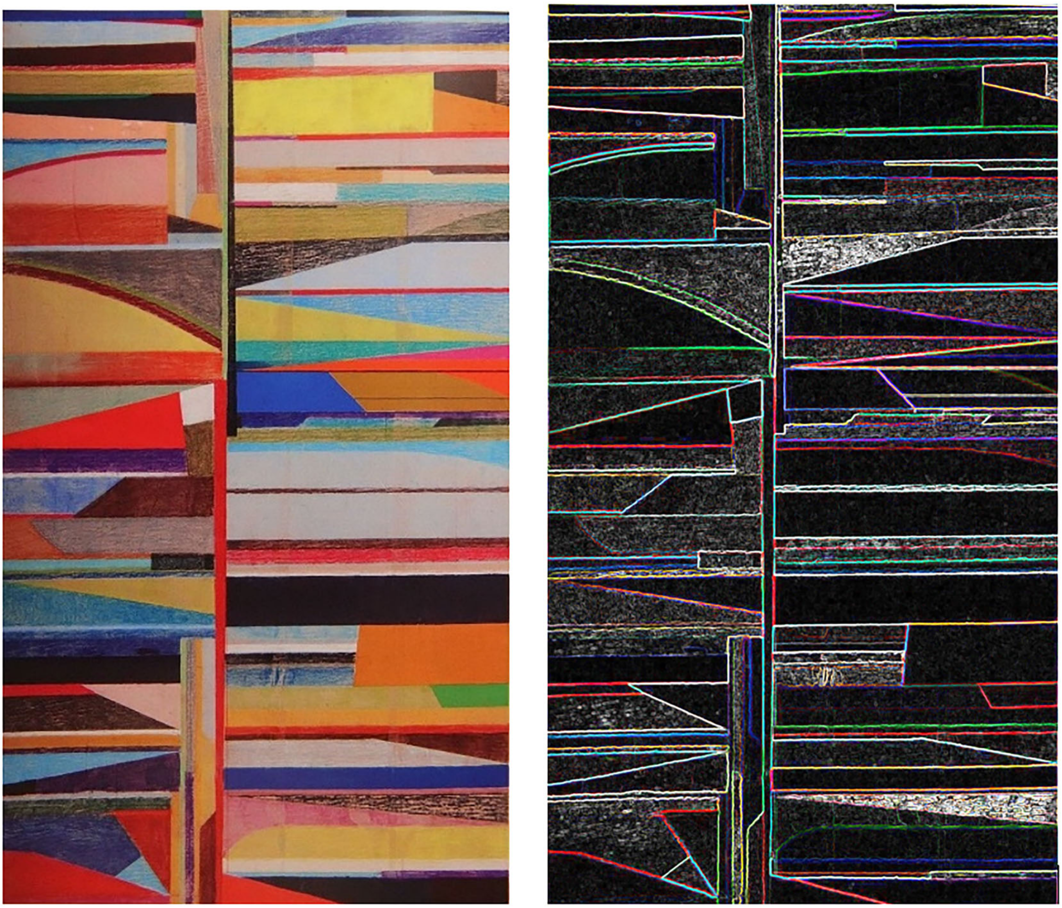


**FIGURE 5** 21st century by Viktor Romanov, with and without a Sobel edge-detection filter, one popular convolutional edge-detection algorithm. (Operation performed by author in Gimp 2.0 Freeware image editing program. Image credit: Yugra News Publishing, Wikimedia Commons CC License)

produces further gains in efficiency, because it greatly reduces the number of parameters that need to be learned, relative to a fully-connected network with the same number of nodes; it also makes computation more efficient when trained networks classify novel images, because the activation functions which need to be computed have far fewer inputs.

### 2.4 | Regularization

All machine learning seeks a tradeoff between underfitting and overfitting training data; training data are underfit if the model does not learn enough structure from the training exemplars to predict its category labels, and overfit if it learns so much idiosyncratic structure from the training data that it fails to generalize to novel exemplars outside the training set. Overfitting is a concern with DCNNs, as analyses have shown that deep networks possess enough storage capacity to simply memorize mappings between exemplars and labels for even very large training corpora, even when exemplars are assigned random labels or consist entirely of random noise (Zhang, Bengio, Hardt, Recht, & Vinyals, 2016).

To avoid overfitting, modelers deploy a variety of explicit regularization techniques. One simple method involves slightly modifying the training exemplars, such as by adding noise or shifting or rotating images. This forces the network to learn that categorizations must be robust to small changes in input details. Another popular method is dropout, which causes some nodes in the network to periodically become inactive, forcing the network's categorizations to not depend too much on any particular regularity (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). A third method modifies the error function to favor simpler or sparser solutions; “$\ell 1$” regularization, for example, adds a penalty term that (roughly) causes link weights to fall back to zero if not maintained by a large gradient during learning (Krizhevsky, Sutskever, & Hinton, 2012). As a result, the performance of the network is biased against learning many precise details, and feature representations tend to become more localized (i.e., depending upon fewer nodes and links) and focused on more generalizable properties. Finally, there are also “implicit” regularization techniques, such as early stopping, which halts learning according to some measure of when generalization improvement starts to plateau and training begins to overfit (Zhang et al., 2016). Recent empirical analyses have suggested that explicit and implicit regularization are requirements for effective DCNN design (Achille & Soatto, 2018).

## 3 | INTERPRETATION AND EXPLANATION

With these basic features of DCNNs in place, we can now ask: Why do these networks tend to work so well? Let us consider three popular explanations for DCNNs' distinctive successes. Though they are sometimes offered as competitors, it is not clear that they are in conflict and may illuminate complementary aspects of the same underlying phenomenon.

### 3.1 | Hierarchical feature composition

The most traditional explanation of the effectiveness of DCNNs is that they work like visual processing has been thought to work in the mammalian ventral stream, by hierarchically composing more complex feature representations from simpler and less abstract ones. The division of labor between convolutional and pooling nodes in early DCNNs was inspired by Hubel and Wiesel's (1967) discovery of two different types of neuron in cat visual cortex: “simple cells,” which seemed to be sensitive to very specific and local features like shadings and contrasts, and “complex cells,” which seemed to take input from a variety of simple cells and detect those features with a bit more invariance to precise location and pose. This story was later extended with a variety of imaging methods to suggest a whole layered processing cascade of hierarchical abstraction in primate visual cortex (Goodale & Milner, 1992), from lines and borders in V5, to angles and colors in TEO/PIT (posterior inferotemporal), and to figures and objects in

TE/AIT (anterior inferotemporal). In this respect, DCNNs and ventral stream processing compare favorably, with both appearing to detect increasingly abstract features as one moves up to later layers of processing (Yamins & DiCarlo, 2016). While more recent critical appraisals have complicated the traditional two-streams story in mammals (e.g., by finding some degree of interaction between the streams, and more recurrent and top-down processing—McIntosh & Schenk, 2009), they have vindicated the role of hierarchical feature composition by highlighting the increasing recoverability of more abstract features as one moves up the ventral stream hierarchy (Hong et al., 2016; Khaligh-Razavi & Kriegeskorte, 2014).

### 3.2 | Systematic transformations of input to adjust for nuisance variation

Another interpretation of DCNNs' success holds their characteristic features to implement a set of unrevisable, domain-general assumptions that help networks control for common forms of variation in perceptual input. Consonant with the previous subsection, depth enforces the assumption that complex features are built from simpler ones. Passing the windows of kernels over the whole image at each layer applies the assumption that each feature can be reused many times in the composition of more complex ones and that features can occur anywhere in the image (e.g., as a hexagon may reuse the same angle feature six times in different locations). Max poolers render the precise pose or location of a feature irrelevant to an exemplar's categorization, because they only pass along activation from their most-activated input filter. For example, if a max pooler took input from several different eye detectors with overlapping spatial sensitivity, it could output only that an eye was detected on approximately the left side of the face, simplifying its exact location; or if the pooler took input from filters which detect lines in different angular rotations but at the same location, it could output only that some line was detected at that location, invariantly of its idiosyncratic pose. Sparse connectivity and regularization impose the assumptions that the recognition of generalizable features should not depend on too many long-distance relations among features (e.g., we do not normally need to look at someone's feet to recognize their facial expression) or subtle contextual details (e.g., we do not normally need to check whether one's facial expression depends upon the weather or the day of the week). During training, DCNNs simply ignore hypotheses that violate these assumptions; but when the solution to a problem satisfies them, DCNNs can find that solution more efficiently than more traditional neural networks (c.f. Bengio et al., 2016, p. 334, for a worked example of how a DCNN can be 60,000 times more efficient in Big-O notation on the task of edge detection).[6]

Thus, perhaps DCNNs work so well because a wide class of classification and decision problems satisfy these assumptions. In particular, machine learning researchers have noted that many visual and auditory tasks are plagued by "nuisance factors": repeatable and systematic sources of variation that are not diagnostic of decision success. For visual classification tasks, common nuisances include size, pose, location, and rotation, or for auditory tasks, pitch, tone, pronunciation, and duration.[7] Decision procedures that succeed on these tasks must learn to systematically adjust for these sources of variance. More cognitive or amodal tasks can fit these assumptions as well; Go strategy, for example, should also be tolerant to small changes in the position or rotation of stone placement patterns, whether board patterns are perceived visually, auditorially, or inputted through amodal symbols. How far the utility of these assumptions extends into territory traditionally ascribed to "higher" cognition remains an open question.

### 3.3 | Number of linear regions

A third, related account of deep learning functionality that is becoming increasingly popular emphasizes the number of linear regions they can map in a problem's input space, relative to networks which do not possess their characteristic features. This idea requires some elaboration. We should first introduce the idea of similarity space: a multidimensional coordinate system in which each dimension marks the degree to which an exemplar exhibits some feature (for a recent philosophical exposition, c.f. Gauker, 2013, pp. 86–93). Exemplars can then be represented as vectors in this feature space, and the perceptual similarity between two exemplars as the distance between their vectors. Categories can moreover be understood as manifolds or regions in this space, where membership in a category is determined by whether the exemplar's vector terminates inside that manifold. Of course, the exact boundaries of each category's manifolds are inaccessible to networks during training; the "goal" of training a neural network for classification can then be understood as discovering a global output function composed of individual nodes' activation functions and associated link weights that can draw boundaries between the manifolds of categories that need to be discriminated. A linear region is, finally, a piece of that output decision function with a linear slope (which can by itself be drawn easily, without consuming much computational resources). The ability to draw more distinct linear regions is advantageous, because it allows neural networks to impose more complex boundaries between categories which are difficult to discriminate from one another because of many low-level perceptual similarities (such as nuisance variation—Figure 6).

A well-known analysis by Montúfar et al. (2014) uses a paper-folding metaphor to explain how the discrimination functions of deep neural networks can have exponentially more linear regions than shallower networks with the same number of nodes. The key is that by finding symmetries in the input and "folding" feature space to align them (by combining a linear and nonlinear activation function, such as rectification and max pooling in DCNNs), these networks can reuse linear regions exponentially many times in terms of the network's depth (c.f. Section 2.1). Deep neural networks thus hierarchically fold input space starting with the first layer of the network, with each successive fold multiplying the number of times that a linear region can be reused in drawing ever-finer discriminations between categories (Figure 7).[8] The learning process can be interpreted as exploring which folds of the feature space exploit symmetries to produce the lowest error functions, while at the same time reducing the computational complexity of the functions that must be computed by later layers. As a result, on problems which benefit from hierarchical transformation, deep networks can be much more efficient than shallower networks that compute the same function. Though some may scoff at these "mere efficiency gains," they would be decisive in evolutionary or behavioral competitions which required making complex decisions under limited time and resources.

This explanation for DCNNs' success is not necessarily incompatible with the first two—hierarchical feature composition and nuisance adjustment may just be two members of a class of tasks that benefit from reuse of linear regions by identifying symmetries in feature space—but there may be others still, giving this third explanation potentially broader applicability.

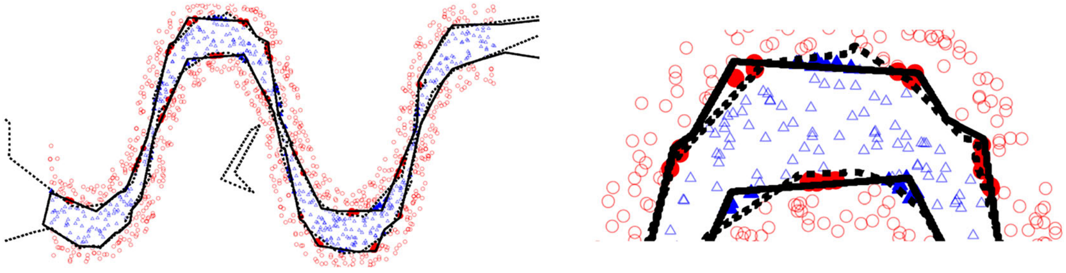


**FIGURE 6** Examples of decisions boundaries from a deep and shallow network on the same input data (reproduced from Montúfar, Pascanu, Cho, & Bengio, 2014). Red dots indicate members of a different category from blue triangles, and the lines (zoomed in on right) indicate decision boundaries drawn between their manifolds by a shallow network (with 20 hidden units, drawing the solid line) or a deep network (with two hidden layers of 10 units each, drawing the dashed line). Filled markers are errors made by the shallow but not the deep network. The straight parts of the lines are their linear regions; each linear region which must be drawn independently consumes its own computational resources

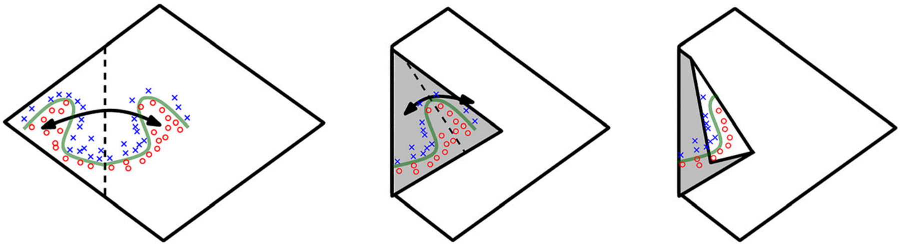


**FIGURE 7** A visual representation of the paper folding metaphor, demonstrating how depth allows linear regions to be reused (from Montúfar et al., 2014). The first fold highlights a symmetry in the discrimination function across the dotted line, allowing the same portion of the discrimination curve to be reused in two different locations in feature space. The second fold identifies a further symmetry, allowing the initial line to be reused four times, and so on exponentially more often in terms of the number of subsequent folds. Because each unique linear section of the discrimination boundary consumes its own computational resources, an exponential reduction in the number of such unique lines which have to be drawn on the folded space brings significant gains in efficiency. (This is the equivalent in DCNNs of the exponential reuse of unique products found in deep sum-product networks, as discussed in Section 2.1)

# 4 | OPEN QUESTIONS, FUTURE RESEARCH DIRECTIONS, AND LIVE DEBATES

Much ink has been spilt prognosticating extreme future trajectories for DCNN-based AI research: either that it has been dramatically oversold and we are about to enter another "AI winter" or that it will soon usher us into a singularity of exponentially-increasing levels of intelligence (Marcus, 2018a; Walsh, 2017). Both of these extreme predictions are probably exaggerated, however, wrapped up as they are with a nebulous and necessarily speculative question about future discoveries. Most commentators agree that current deep learning methods fall short of implementing general intelligence, and it remains an open question as to whether some modification of current deep learning methods will be able to do so. Even if the answer to this question is negative; however, I have recently argued that current DCNNs already model a crucially important component of mammalian intelligence—a particular kind of abstraction—that until now eluded our grasp (Buckner, 2018), and many pressing and more answerable questions still concern the proper interpretation and implications of established DCNN methods. I end by canvassing three related lines of inquiry that could be profitably explored by philosophers of mind and cognitive science in the near future.[9]

## 4.1 | DCNNs and nativism/empiricism

DCNN modelers have often made ambitious claims about the relevance of their research to old debates between empiricism and nativism in philosophy of mind, sometimes citing the work of classical empiricists like John Locke or David Hume (Silver et al., 2017). These claims intersect with conversations from the Golden Age of connectionism, as philosophers appreciated early on that neural networks might provide a proof of concept that certain types of knowledge could be learned directly from sensory experience, without innate knowledge (c.f. Bechtel & Abrahamsen, 2002, pp. 54–57). Connectionist research was thus often associated with the banner of classical empiricism, and classical rule-and-symbol based methods with the banner of rationalism. It was also soon appreciated, however, that neural network research could be rationalist in orientation, in the sense that innate knowledge could be modeled by presetting link weights in a way that biased networks to acquire particular information (Elman, Bates, & Johnson, 1998). Some of the highest-profile recent success of DCNNs, however—most especially AlphaZero, which can defeat human grandmasters and other top-ranked artificial systems in chess, shoji, and Go by learning strategies entirely from self-play—were claimed to strongly vindicate the empiricist approach (Silver et al., 2017). This latter claim has in turn been challenged by rationalists, who note that AlphaZero is in some sense not a pure "blank slate," building in significant knowledge about the rules of Go and mechanisms that systematically explore possible outcomes one at a time, such as Monte Carlo Tree Search (Marcus, 2018b). Some prominent deep learning researchers have also called for a return to innate biases in deep networks to help them master tasks like causal reasoning or linguistic processing (Battaglia et al., 2018).

This is a place where philosophers could productively engage with active debates in artificial intelligence, for the proponents and critics of AlphaZero seem to be talking past one another by operating on different conceptions of the empiricist/nativist divide. Marcus in particular argues that only a system based on a very extreme interpretation of a "blank slate" is worth of the empiricist title, thinking of cognition as ranging over four variables, *a* (algorithms), *r* (representational formats), *k* (innate knowledge), and *e* (experience). As Marcus puts it—and in public interactions, some deep learning pioneers like LeCun seem to agree—a truly empiricist system should begin with nothing for variables *r* and *k*, and only a very minimal amount of structure in *a*, deriving everything else from *e*. If this were right, then AlphaZero should be quite far from a purely empiricist system, as it both deploys tree search algorithms (which it did not learn from experience), significant forms of domain-general knowledge pertaining to nuisance variables that we discussed in Section 3.2, and the rules of Go (the last of which its builders concede is "innate" in the relevant sense). In related debates, however, other prominent rationalists have disagreed that this is a useful construal of empiricism, noting that it has long been true that without memory, attention, and a variety of other domain-general faculties, no system could learn anything from any amount of experience (Laurence & Margolis, 2015).

If empiricism is to be more than a strawman, then the debate might more profitably center on whether the innate machinery which must be built-in to deep networks is domain-specific or domain-general. Even on this more modest construal, there remains an interesting and lively debate that empiricists and nativists could explore through deep neural network research, specifically regarding how much human knowledge can be learned using only domain-general faculties and assumptions (such as those highlighted in Section 3.2). This construal would pit much deep learning research against powerful nativist contenders in cognitive science, such as the core cognition program (which argues that at least a few domain-specific core concepts like OBJECT, AGENT, NUMBER, or CAUSE are innate—Carey, 2009) or nativist versions of Bayesian cognitive modeling (which can deploy domain-specific learning rules, representational primitives, or prior probability estimations—c.f. Colombo, 2018; Perfors, 2012). DCNNs may also distinctively address some long-standing mysteries in empiricist philosophy of mind regarding the role of abstraction in the acquisition of categorical knowledge, a faculty that has frequently been invoked by empiricists to account for learners' transition from basic sensory associations to the abstract representations deployed in higher cognition but has until recently not been adequately explained (c.f. Buckner, 2018; Laurence & Margolis, 2012).

## 4.2 | But do DCNNs learn the way that humans do?

Even if DCNNs are thought to have provided a proof of concept that substantial amounts of abstract human knowledge can be learned without domain-specific resources, there may be some doubt as to whether they demonstrate that this is how humans and animals actually acquire this knowledge (Lake, Ullman, Tenenbaum, & Gershman, 2016; Marcus, 2018a). This concern can take many forms, but I will focus on two of the most pressing here. First, critics have worried that the most successful DCNNs require far more training exemplars—especially supervised learning on large training sets, where the correct answers are labeled—than humans and animals require to learn the same information. Second, they have argued that the phenomenon of "adversarial examples" (a concept which may not be possible to define except by ostension, for reasons to be elaborated below) demonstrates that what is learned by DCNNs differs substantially from what is learned by humans and animals.

Most successful DCNNs require millions of training exemplars to reach their benchmark performance. The standard methods of training image-labelling DCNNs, for example, involves supervised backpropagation learning using the ImageNet database, which contains 14 million images that are hand annotated with labels from over 20,000 object categories. To consider another example, AlphaGo's networks were trained on over 160,000 stored Go games recorded from human grandmaster play and then further learned by playing millions of games against iteratively stronger versions of itself (over 100 million matches in total); its human opponent Lee, by contrast, could not have played more than 50,000 matches in his entire life. Skeptics thus wonder whether deep neural networks will ever be able to learn from smaller, more human-like amounts of experience (at least, without integrating more domain-specific structures or knowledge).

These concerns might be tempered, however, with more careful reflection upon how to score human performance fairly in such comparisons, and by modeling more domain-general faculties in DCNN-based systems so that they can exploit a smaller number of training instances more effectively. Two factors are often neglected in counting the number of exemplars that humans should be scored as having been exposed to in learning: (a) that many different vantage points of the same object could provide additional training exemplars for cortical learning and (b) that offline memory consolidation during sleep and daydreaming can replay the same training session many thousands of times in nonconscious repetitions. We already know that when deep learning models are trained on successive frames from video rather than static exemplars, many different vantage points of the same object can be treated as thousands of independent training instances (Luc, Neverova, Couprie, Verbeek, & LeCun, 2017). Furthermore, if models are supplemented with "episodic replay" buffers that are inspired by domain-general memory faculties in mammals, a DCNN's performance can continue to benefit from repeatedly replaying exposure to the same training instances numerous times (Blundell et al., 2016; Mnih et al., 2015; Vinyals, Blundell, Lillicrap, Kavukcuoglu, & Wierstra, 2016).[10] Under the reasonable estimates that human perception has a frame rate of 10–12 images per second, and memories may be repeatedly reconsolidated by theta rhythm hundreds of times over a period of months and years, an infant's 10-min interaction with a new toy might be fairly scored as providing tens of thousands of trainable exposures, rather than a single one, as common sense might suppose.

Even if DCNNs can be made to learn from more human-sized training sets through such methods, however, a more robust and perplexing problem remains, arising from so-called "adversarial examples" (Szegedy et al., 2013). Adversarial examples were originally defined as images created by slightly modifying an easily-classifiable exemplar in a way that was imperceptible to humans, but which could cause dramatic misclassification by DCNNs by maximizing their prediction error (Figure 8); this first kind of adversarial example has come to be called a "perturbed image." Perturbed images could sometimes be overcome by simply de-noising or rescaling images; but a second family of approaches was soon discovered, which relied on the use of nonsense patterns or juxtapositions—so-called "fooling" or "rubbish" images—to produce highly confident (mis)classifications in DCNNs that were more resistant to simple countermeasures. Subsequent research has found that adversarial examples have many counterintuitive properties: they can transfer with labels to other DCNNs with different architectures and training sets, they are difficult to distinguish from real exemplars using preprocessing methods, and they can be created without "god's-eye" access to

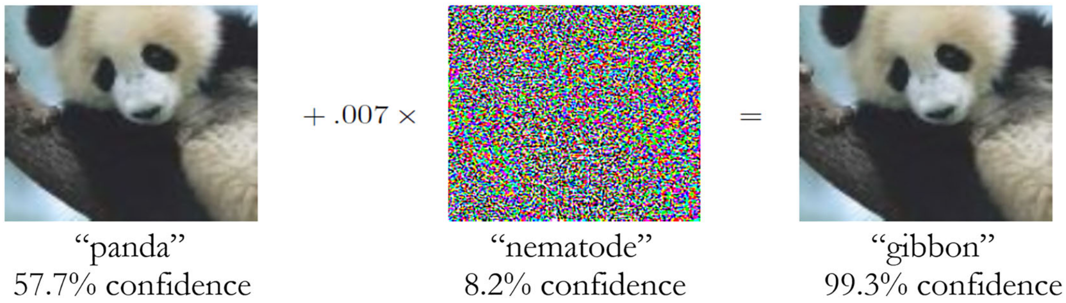


**FIGURE 8** An adversarial "perturbed image" reproduced from Goodfellow, Shlens, and Szegedy (2014). After the "panda" image was modified slightly by the addition of a small noise vector (itself classified with low confidence as a nematode), it was classified as a gibbon with high confidence, despite the modification being imperceptible to humans

model parameters or training data. Rather than being an easily overcome quirk of particular models or training sets, they appear to highlight a robust property of current DCNN methods.

Much of the interest of adversarial examples has derived from the assumption that humans do not see them as DCNNs do. For practical purposes, this would entail that hackers and other malicious agents could use them to fool automated vision systems—for example, by placing a decal on a stop sign that caused an automated vehicle to classify it as a yield sign—and human observers might not know that anything was awry until it was too late. For modeling purposes, however, they might also show that despite categorizing naturally-occurring images as well or better than human adults, DCNNs do not really acquire the same kind of category knowledge that humans do—perhaps instead building “a Potemkin village that works well on naturally occurring data, but is exposed as fake when one visits points in space that do not have a high probability” (Goodfellow, Shlens, & Szegedy, 2014).

However, more recent investigations have challenged the assumption that a DCNN's take on adversarial examples is really so alien to human perception, either by producing perturbed images that fool humans (Figure 9, Elsayed et al., 2018) or by showing that humans can easily “adopt the machine perspective” and, when forced to choose between a preset list of candidate labels, predict a DCNN's labels for adversarial examples with high accuracy (Figure 10, Zhou & Firestone, 2019). These authors suggest that the behavior of DCNNs in these cases might be due to the fact that the final classification layer always forces them to choose among a list of candidate labels, even when images are very different from previously-classified exemplars. Furthermore, it may be that DCNNs do in fact capture some aspects of lower-level perceptual categorization in humans, such that adversarial examples do *look like* members of the purportedly incorrect label class in some sense, even if humans do not ultimately think that they *actually are* members of that class. DCNNs may thus succeed in modeling human perceptual similarity judgments but not yet have the resources to draw a distinction between what something *looks like* and what it *really is.* If such comparisons to humans are to be meaningful in the future, more care must be put into defining key research terms—in particular, by not making it true by definition that adversarial examples cannot fool humans—and by scrutinizing task characteristics so that comparisons between humans and machines can be fairly investigated using empirical methods. As it stands, adversarial examples and their implications remain mysterious and would benefit from further philosophical reflection.

## 4.3 | What kind of explanation do DCNNs provide?

Even if it is conceded that DCNNs do explain human perceptual similarity judgments, there will remain significant debate in philosophy of science as to what kind of explanation they provide. Much of the enthusiasm for neural

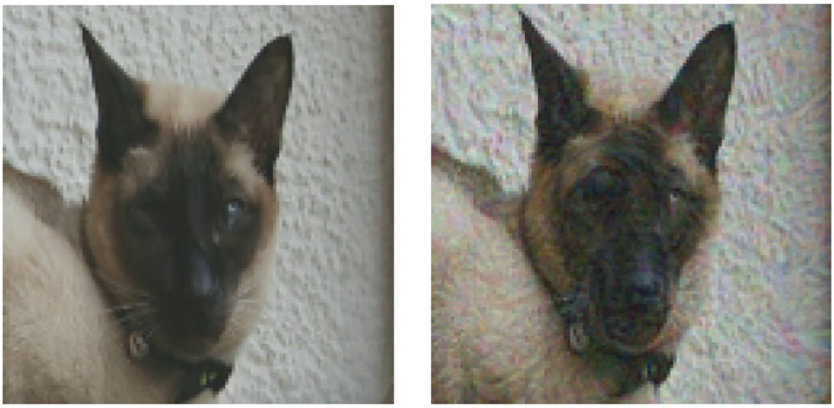


**FIGURE 9** A perturbed image that can purportedly fool human subjects, with the original image of a cat on the left, and the perturbed image (often classified as a dog) on the right. (Image reproduced from Elsayed et al., 2018)

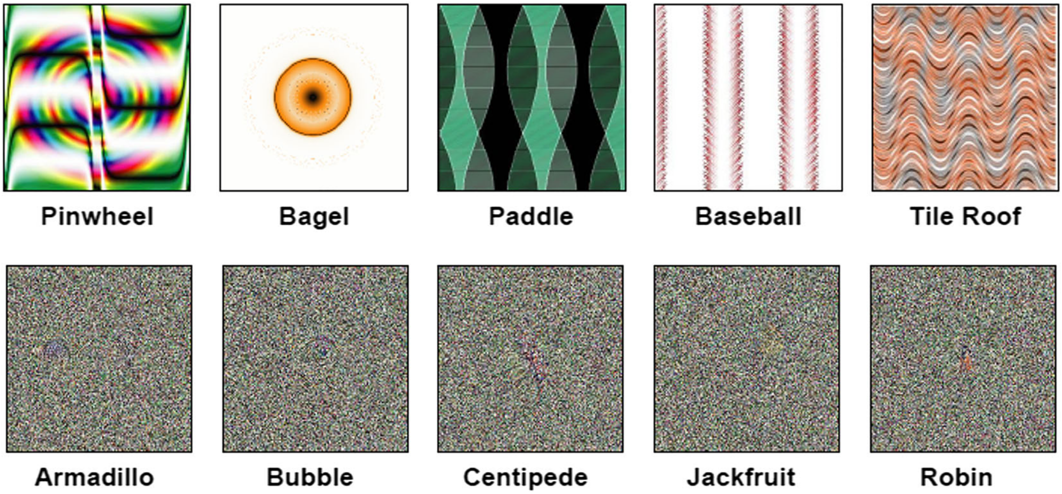


**FIGURE 10** Examples of two different types of rubbish images tested by Zhou and Firestone (2019) with preferred DCNN labels. In a forced-choice task, humans were able to guess a DCNN's preferred labels for these images with high accuracy

networks in the past has been derived from their supposed structural similarity to brains; a wave of “new mechanism” cresting in philosophy of science (Glennan, 2017) may seem well placed to vindicate this enthusiasm with a rigorous model of explanation. This movement holds (roughly) that explanations in the life sciences work by locating structures and their organization in a target system whose coordinated operations regularly produce the phenomenon of interest. DCNNs, however, are pitched at a high level of abstraction from perceptual cortex, which could lead one to doubt that they succeed at providing mechanistic explanations for even perceptual processing (though see Boone & Piccinini, 2016; Stinson, 2018). In particular, some key aspects of the Hubel and Wiesel's story which originally inspired DCNNs have been challenged on neuroanatomical grounds (especially regarding the purported dichotomy between simple and complex cells on which the cooperation between convolution and pooling was based—Priebe, Mechler, Carandini, & Ferster, 2004), and there remains significant debate as to whether backpropagation learning is biologically plausible (though more biologically-plausible learning algorithms have recently been explored by prominent deep learning modelers, especially involving randomized error signals and spike-timing dependent plasticity—Lillicrap, Cownden, Tweed, & Akerman, 2016; Scellier & Bengio, 2017).

A backlash against “mechanistic imperialism” has begun highlighting a role for functional, fictional, noncausal, and mathematical explanations in the life sciences (Weiskopf, 2011) and indeed in computational neuroscience more specifically (Chirimuuta, 2018). These authors are likely to think that the attempt to force DCNN-based explanations into a mechanistic mold is a mistake, emphasizing instead the computational or mathematical principles which could be implemented in a variety of very different mechanisms. In particular, they may focus on the appeals to mathematical efficiency that Section 3 provided in explaining DCNNs' successes. Mechanists are in turn likely to respond by noting that while the abstract mathematical aspects of these explanations are important, paradigm mechanistic explanations often involved mathematical formalism; and if DCNNs are to provide more than a “how possibly” story concerning human cortex's ability to implement these forms of processing, we must identify some structural or causal correspondences that show this is how they actually do it (Piccinini & Craver, 2011). DCNNs can thus serve as an important test case in the debate between mechanists and non-mechanists—though both sides should attend carefully to the details of these models and their popular analyses, canvassed here.

Thus ends a list of open questions; let us waste no more time in attempting to answer them.

## ACKNOWLEDGEMENTS

I thank Zed Adams, Colin Allen, David Chalmers, Mazviita Chirimuuta, James Garson, Hajo Greif, Edouard Machery, J. Brendan Ritchie, Anna-Mari Rusanen, Bruce Rushing, Carlos Zednik, an anonymous reviewer for this journal, and audiences at the Technical University of Munich, the German Society for the Philosophy of Science, and the University of Cambridge for comments and questions on earlier drafts. I also thank the Leverhulme Trust for their generous funding of the Leverhulme Centre for the Future of Intelligence, which supported me with a visiting fellowship at the University of Cambridge during the revision of this article.

## ENDNOTES

[1] A critical concern, however, is that these results are often not easily reproducible and may depend upon underreported model parameters or idiosyncratic aspects of training (Henderson et al., 2018).

[2] The evidence that DCNNs are good models of perceptual similarity comes from many directions. Perhaps the strongest lines of evidence come from comparing representational similarity judgments of humans and DCNNs directly (Khaligh-Razavi & Kriegeskorte, 2014) and electrophysiological recordings from visual cortex to activation patterns of layers of a deep network, which find strong correlations between the recoverability of features at various stages of the visual cortex and DCNNs (Yamins & DiCarlo, 2016; Hong, Yamins, Majaj, & DiCarlo, 2016; though see also Rajalingham et al., 2018). Other studies have shown that DCNNs exhibit some of the same learning biases as humans and animals (Ritter, Barrett, Santoro, & Botvinick, 2017).

[3] Most recently, this trend is beginning to change. Since the initial submission of this manuscript, several articles written by philosophers and directly engaging with deep learning were either just published or are about to be published (see Bringsjord, Govindarajulu, Banerjee, & Hummel, 2018; Buckner, 2018; Floridi, 2019; López-Rubio, 2018; Miracchi, 2019; Schubbach, 2019; Shevlin & Halina, 2019; Zednik, 2019).

[4] Most generally, convolution is a mathematical operation on two functions that produces a third function expressing how the shape of the first is modified by the second. Most DCNNs deploy a version of discrete convolution where the first input is a multidimensional array of input data and the second input is a learned, multidimensional array of parameters that, when applied to the input array in an operation akin to matrix multiplication, amplifies the presence of certain features and minimizes the presence of others in its output (a third, transformed array, often called a "feature map"—see Figure 3). For a longer discussion, see Bengio, Courville, and Goodfellow (2016, pp. 327–329).

[5] Downsampling is an operation which selects a subset of input data to build a compressed representation which preserves important information from the original. For example, image compression in the .jpg or .gif file formats involves downsampling, as does sound compression in .mp3s.

[6] Big-O notation is a mathematical formalism used in computer science to classify algorithms by how their running time or space requirements grow as the size of their input increases.

[7] More precisely, DCNNs are thought efficient at overcoming nuisance parameters with a group-like structure—that is, which adhere to the axioms of group theory and thus afford the use of systematic geometric operations such as scaling or affine transformation to discover invariant properties (Achille & Soatto, 2018; Anselmi, Rosasco, & Poggio, 2016). Non-group-like nuisance parameters include occlusion or changes in illumination, on which DCNNs may be no more efficient than alternative neural network architectures.

[8] The feature spaces, of course, have many more than two dimensions, and the "folds" can often involve dimensionality reduction; but the basic point stands.

[9] Many interesting questions remain regarding DCNNs in applied ethics and philosophy of law, but we do not canvass them here. For some reviews of recent concerns, see Lin, Abney, and Jenkins (2017) and Zednik (2019).

[10] Further reductions in the number of training exemplars required have been achieved by adding other biologically-inspired, domain-general components, such as attention and reinforcement learning; for a review, see Hassabis, Kumaran, Summerfield, and Botvinick (2017).

## ORCID

*Cameron Buckner* https://orcid.org/0000-0003-0611-5354

## WORKS CITED

Achille, A., & Soatto, S. (2018). Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, *19*(50), 1–34.

Anselmi, F., Rosasco, L., & Poggio, T. (2016). On invariance and selectivity in representation learning. *Information and Inference*, *5*(2), 134–158. https://doi.org/10.1093/imaiai/iaw009

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., … Faulkner, R. (2018). Relational inductive biases, deep learning, and graph networks. ArXiv Preprint ArXiv:1806.*01261*.

Bechtel, W., & Abrahamsen, A. (2002). *Connectionism and the mind: Parallel processing, dynamics, and evolution in networks*. Oxford, UK: Blackwell Publishing.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends® in Machine Learning*, *2*(1), 1–127. https://doi.org/10.1561/2200000006

Bengio, Y., Courville, A., & Goodfellow, I. (2016). *Deep learning*. Boston: MIT Press.

Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J. Z., … Hassabis, D. (2016). Model-free episodic control. ArXiv Preprint ArXiv:1606.*04460*.

Boone, W., & Piccinini, G. (2016). Mechanistic abstraction. *Philosophy of Science*, *83*(5), 686–697. https://doi.org/10.1086/687855

Bringsjord, S., Govindarajulu, N. S., Banerjee, S., & Hummel, J. (2018). Do machine-learning machines learn? In V. C. Müller (Ed.), *Philosophy and theory of artificial intelligence 2017* (pp. 136–157). Cham, Switzerland: Springer International Publishing. https://doi.org/10.1007/978-3-319-96448-5_14

Buckner, C. (2018). Empiricism without magic: Transformational abstraction in deep convolutional neural networks. *Synthese*, *195*(12), 5339–5372. https://doi.org/10.1007/s11229-018-01949-1

Buckner, C., & Garson, J. (2018). Connectionism and post-connectionist models. In M. Sprevak, & M. Columbo (Eds.), *The Routledge handbook of the computational mind* (pp. 76–91). London: Routledge University Press.

Carey, S. (2009). *The origin of concepts*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195367638.001.0001

Chirimuuta, M. (2018). Explanation in computational neuroscience: Causal and non-causal. *The British Journal for the Philosophy of Science*, *69*(3), 849–880. https://doi.org/10.1093/bjps/axw034

Colombo, M. (2018). Bayesian cognitive science, predictive brains, and the nativism debate. *Synthese*, *195*(11), 4817–4838. https://doi.org/10.1007/s11229-017-1427-7

Delalleau, O., & Bengio, Y. (2011). Shallow vs. deep sum-product networks. In *Advances in neural information processing systems* (pp. 666–674). http://papers.nips.cc/paper/4350-shallow-vs-deep-sum-product-networks: http://papers.nips.cc/paper/4350-shallow-vs-deep-sum-product-networks.

Elman, J. L., Bates, E. A., & Johnson, M. H. (1998). *Rethinking innateness: A connectionist perspective on development* (Vol. 10). Cambridge, MA: MIT press.

Elsayed, G., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I., & Sohl-Dickstein, J. (2018). *Adversarial Examples that Fool both Computer Vision and Time-Limited Humans, Advances in Neural Information Processing Systems*, 31: 3910–3920.

Floridi, L. (2019). What the near future of artificial intelligence could be. *Philosophy & Technology*, *32*(1), 1–15. https://doi.org/10.1007/s13347-019-00345-y

Gauker, C. (2013). *Words and images: An essay on the origin of ideas* (Reprint ed.). Oxford: Oxford University Press.

Glennan, S. (2017). *The new mechanical philosophy*. London: Oxford University Press.

Goodale, M. A., & Milner, A. D. (1992). Separate visual pathways for perception and action. *Trends in Neurosciences*, *15*(1), 20–25. https://doi.org/10.1016/0166-2236(92)90344-8

Goodfellow, I., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. ArXiv Preprint ArXiv: 1412.*6572*.

Hassabis, D., Kumaran, D., Summerfield, C., & Botvinick, M. (2017). Neuroscience-inspired artificial intelligence. *Neuron*, *95* (2), 245–258. https://doi.org/10.1016/j.neuron.2017.06.011

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. *Thirty-Second AAAI Conference on Artificial Intelligence.*

Hong, H., Yamins, D. L., Majaj, N. J., & DiCarlo, J. J. (2016). Explicit information for category-orthogonal object properties increases along the ventral stream. *Nature Neuroscience*, *19*(4), 613–622. https://doi.org/10.1038/nn.4247

Hubel, D. H., & Wiesel, T. N. (1967). Cortical and callosal connections concerned with the vertical meridian of visual fields in the cat. *Journal of Neurophysiology*, *30*(6), 1561–1573. https://doi.org/10.1152/jn.1967.30.6.1561

Khaligh-Razavi, S.-M., & Kriegeskorte, N. (2014). Deep supervised, but not unsupervised, models may explain IT cortical representation. *PLoS Computational Biology*, *10*(11), e1003915. https://doi.org/10.1371/journal.pcbi.1003915

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097–1105. Retrieved from http://papers.nips.cc/paper/4824-imagenet-classification-w

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2016). Building machines that learn and think like people. *Behavioral and Brain Sciences*, *40*, 1–101. https://doi.org/10.1017/S0140525X16001837

Laurence, S., & Margolis, E. (2012). Abstraction and the origin of general ideas. Philosopher's Imprint, 12(19). Retrieved from http://hdl.handle.net/2027/spo.3521354.0012.019

Laurence, S., & Margolis, E. (2015). Concept nativism and neural plasticity. In S. Laurence, & E. Margolis (Eds.), *The conceptual mind: New directions in the study of concepts* (pp. 117–147). Cambridge, MA: MIT Press.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. *Advances in Neural Information Processing Systems*, *https://papers.nips.cc/paper/293-handwritten-digit-recognitionwith-a-back-propagation-network*, 396–404.

Lillicrap, T. P., Cownden, D., Tweed, D. B., & Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, *7*. https://doi.org/10.1038/ncomms13276

Lin, P., Abney, K., & Jenkins, R. (2017). *Robot ethics 2.0: From autonomous cars to artificial intelligence*. Oxford, UK: Oxford University Press.

López-Rubio, E. (2018). Computational functionalism for the deep learning era. *Minds and Machines*, *28*(4), 667–688. https://doi.org/10.1007/s11023-018-9480-7

Luc, P., Neverova, N., Couprie, C., Verbeek, J., & LeCun, Y. (2017). Predicting deeper into the future of semantic segmentation. IEEE international conference on computer vision (ICCV), *1*.

Makridakis, S. (2017). The forthcoming artificial intelligence (AI) revolution: Its impact on society and firms. *Futures*, *90*, 46–60. https://doi.org/10.1016/j.futures.2017.03.006

Marcus, G. (2018a). Deep learning: A critical appraisal. ArXiv Preprint ArXiv:1801.*00631*.

Marcus, G. (2018b). Innateness, AlphaZero, and artificial intelligence. ArXiv Preprint ArXiv:1801.*05667*.

McIntosh, R. D., & Schenk, T. (2009). Two visual streams for perception and action: Current trends. *Neuropsychologia*, *47*(6), 1391–1396. https://doi.org/10.1016/j.neuropsychologia.2009.02.009

Minsky, M., & Papert, S. A. (1969). *Perceptrons: An introduction to computational geometry*. Cambridge, MA: MIT press.

Miracchi, L. (2019). A competence framework for artificial intelligence research. *Philosophical Psychology*, *32*, 589–634. https://doi.org/10.1080/09515089.2019.1607692

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., … Ostrovski, G. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

Montúfar, G. F., Pascanu, R., Cho, K., & Bengio, Y. (2014). On the number of linear regions of deep neural networks. *Advances in Neural Information Processing Systems*, *On NIPS website*, *http://papers.nips.cc/paper/5422-on-the-number-of-linear-regions-of*, 2924–2932.

Perfors, A. (2012). Bayesian models of cognition: What's built in after all? *Philosophy Compass*, *7*(2), 127–138. https://doi.org/10.1111/j.1747-9991.2011.00467.x

Piccinini, G., & Craver, C. (2011). Integrating psychology and neuroscience: Functional analyses as mechanism sketches. *Synthese*, *183*(3), 283–311. https://doi.org/10.1007/s11229-011-9898-4

Priebe, N. J., Mechler, F., Carandini, M., & Ferster, D. (2004). The contribution of spike threshold to the dichotomy of cortical simple and complex cells. *Nature Neuroscience*, *7*(10), 1113–1122. https://doi.org/10.1038/nn1310

Rajalingham, R., Issa, E. B., Bashivan, P., Kar, K., Schmidt, K., & DiCarlo, J. J. (2018). Large-scale, high-resolution comparison of the core visual object recognition behavior of humans, monkeys, and state-of-the-art deep artificial neural networks. *Journal of Neuroscience*, *38*(33), 7255–7269. https://doi.org/10.1523/JNEUROSCI.0388-18.2018

Ritter, S., Barrett, D. G., Santoro, A., & Botvinick, M. M. (2017). Cognitive psychology for deep neural networks: A shape bias case study. ArXiv Preprint ArXiv:1706.*08606*. Retrieved from https://arxiv.org/abs/1706.08606

Scellier, B., & Bengio, Y. (2017). Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in Computational Neuroscience*, *11*, 11. https://doi.org/10.3389/fncom.2017.00024

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Schubbach, A. (2019). Judging machines: Philosophical aspects of deep learning. *Synthese*, *still in "First Online" status*, 1–21. https://doi.org/10.1007/s11229-019-02167-z

Shevlin, H., & Halina, M. (2019). *Apply rich psychological terms in AI with care*. *Nature Machine Intelligence* (Vol. 1) (pp. 165–167). https://doi.org/10.1038/s42256-019-0039-y

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., … Graepel, T. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, *362*(6419), 1140–1144. https://doi.org/10.1126/science.aar6404

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., … Bolton, A. (2017). Mastering the game of Go without human knowledge. *Nature*, *550*(7676), 354–359. https://doi.org/10.1038/nature24270

Singhal, H. (2017, June 17). Convolutional neural network with TensorFlow implementation. Retrieved January 30, 2019, from Medium website: https://medium.com/data-science-group-iitr/building-a-convolutional-neural-network-in-python-with-tensorflow-d251c3ca8117

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Stinson, C. (2018). Explanation and connectionist models. In M. Sprevak, & M. Colombo (Eds.), *The Routledge handbook of the computational mind* (pp. 120–134). New York, NY: Routledge.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2013). Intriguing properties of neural networks. *ArXiv:1312.6199 [Cs]*. Retrieved from http://arxiv.org/abs/1312.6199

Vinyals, O., Babuschkin, I., Chung, J., Mathieu, M., Jaderberg, M., Czarnecki, W. (2019). AlphaStar: Mastering the real-time strategy game StarCraft II [Blog]. Retrieved April 5, 2019, from DeepMind website: https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in neural information processing systems 29* (pp. 3630–3638). Retrieved from http://papers.nips.cc/paper/6385-matching-networks-for-one-shot-learning.pdf

Walsh, T. (2017). The singularity may never be near. *AI Magazine*, *38*(3), 58–62. https://doi.org/10.1609/aimag.v38i3.2702

Weiskopf, D. (2011). Models and mechanisms in psychological explanation. *Synthese*, *183*(3), 313–338. https://doi.org/10.1007/s11229-011-9958-9

Yamins, D. L., & DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nature Neuroscience*, *19*(3), 356–365. https://doi.org/10.1038/nn.4244

Zednik, C. (2019). Solving the black box problem: A general-purpose recipe for explainable artificial intelligence. *ArXiv:1903.04361 [Cs]*. Retrieved from http://arxiv.org/abs/1903.04361

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. ArXiv Preprint ArXiv:1611*.03530.*

Zhou, Z., & Firestone, C. (2019). Humans can decipher adversarial images. *Nature Communications*, *10*(1), 1334. https://doi.org/10.1038/s41467-019-08931-6

## AUTHOR BIOGRAPHY

**Cameron Buckner** is an Assistant Professor in the Department of Philosophy at the University of Houston. He began his academic career in logic-based artificial intelligence. This research inspired an interest into the relationship between classical models of reasoning and the (usually very different) ways that humans and animals actually solve problems, which led him to the discipline of philosophy. He received a PhD in Philosophy at Indiana University in 2011 and an Alexander von Humboldt Postdoctoral Fellowship at Ruhr-University Bochum from 2011 to 2013. His research interests lie at the intersection of philosophy of mind, philosophy of science, animal cognition, and artificial intelligence, and he teaches classes on all these topics. Recent representative publications include "Empiricism without Magic: Transformational Abstraction in Deep Convolutional Neural Networks" (2018, *Synthese*), and "Rational Inference: The Lowest Bounds" (2017, *Philosophy and Phenomenological Research*)—the latter of which won the American Philosophical Association's Article Prize for the period of 2016–2018.

**How to cite this article:** Buckner C. Deep learning: A philosophical introduction. *Philosophy Compass*. 2019; 14:e12625. https://doi.org/10.1111/phc3.12625