REVIEW
published: 05 January 2021
doi: 10.3389/fpsyg.2020.513474

Artificial Intelligence Is Stupid and Causal Reasoning Will Not Fix It

*J. Mark Bishop**

Department of Computing, Goldsmiths, University of London, London, United Kingdom

Artificial Neural Networks have reached "grandmaster" and even "super-human" performance across a variety of games, from those involving perfect information, such as Go, to those involving imperfect information, such as "Starcraft". Such technological developments from artificial intelligence (AI) labs have ushered concomitant applications across the world of business, where an "AI" brand-tag is quickly becoming ubiquitous. A corollary of such widespread commercial deployment is that when AI gets things wrong—an autonomous vehicle crashes, a chatbot exhibits "racist" behavior, automated credit-scoring processes "discriminate" on gender, etc.—there are often significant financial, legal, and brand consequences, and the incident becomes major news. As Judea Pearl sees it, the underlying reason for such mistakes is that "*... all the impressive achievements of deep learning amount to just curve fitting*." The key, as Pearl suggests, is to replace "reasoning by association" with "causal reasoning" —the ability to infer causes from observed phenomena. It is a point that was echoed by Gary Marcus and Ernest Davis in a recent piece for the *New York Times*: "*we need to stop building computer systems that merely get better and better at detecting statistical patterns in data sets—often using an approach known as 'Deep Learning'—and start building computer systems that from the moment of their assembly innately grasp three basic concepts: time, space, and causality*." In this paper, foregrounding what in 1949 Gilbert Ryle termed "a category mistake", I will offer an alternative explanation for AI errors; it is not so much that AI machinery cannot "grasp" causality, but that AI machinery (qua computation) cannot understand anything at all.

Keywords: dancing with pixies, Penrose-Lucas argument, causal cognition, artificial neural networks, artificial intelligence, cognitive science, Chinese room argument

OPEN ACCESS

Edited by:
Andrew Tolmie,
University College London,
United Kingdom

Reviewed by:
Manuel Bedia,
University of Zaragoza, Spain
Wolfgang Schoppek,
University of Bayreuth, Germany

****Correspondence:***
J. Mark Bishop
m.bishop@gold.ac.uk

Specialty section:
This article was submitted to
Cognitive Science,
a section of the journal
Frontiers in Psychology

Received: *20 November 2019*
Accepted: *07 September 2020*
Published: *05 January 2021*

Citation:
Bishop JM (2021) Artificial Intelligence Is Stupid and Causal Reasoning Will Not Fix It. Front. Psychol. 11:513474. doi: 10.3389/fpsyg.2020.513474

1. MAKING A MIND

For much of the twentieth century, the dominant cognitive paradigm identified the mind with the brain; as the Nobel laureate Francis Crick eloquently summarized:

> "You, your joys and your sorrows, your memories and your ambitions, your sense of personal identity and free will, are in fact no more than the behavior of a vast assembly of nerve cells and their associated molecules. As Lewis Carroll's Alice might have phrased, 'You're nothing but a pack of neurons'. This hypothesis is so alien to the ideas of most people today that it can truly be called astonishing" (Crick, 1994).

Motivation for the belief that a computational simulation of the mind is possible stemmed initially from the work of Turing (1937) and Church (1936) and the "Church-Turing hypothesis"; in Turing's formulation, every "function which would naturally be regarded as computable" can be computed by the "Universal Turing Machine." If computers can adequately model the brain, then, theory goes, it ought to be possible to *program* them to act like minds. As a consequence, in the latter part of the twentieth century, Crick's "Astonishing Hypothesis" helped fuel an explosion of interest in connectionism: both high-fidelity simulations of the brain (computational neuroscience; theoretical neurobiology) and looser—merely "neural inspired" —analoges (cf. Artificial Neural Networks, Multi-Layer Perceptrons, and "Deep Learning" systems).

But the fundamental question that Crick's hypothesis raises is, of course, that if we ever succeed in fully instantiating a *sufficiently accurate* simulation of the brain on a digital computer, will we also have fully instantiated a digital [computational] mind, with all the human mind's causal power of teleology, understanding, and reasoning, and will artificial intelligence (AI) finally have succeeded in delivering "Strong AI"[1].

Of course, *if* strong AI is possible, accelerating progress in its underpinning technologies[2]–entailed both by the use of AI systems to design ever more sophisticated AIs and the continued doubling of raw computational power every 2 years[3]—will eventually cause a runaway effect whereby the AI will inexorably come to exceed human performance on all tasks[4]; the so-called point of [technological] "singularity" ([in]famously predicted by Ray Kurzweil to occur as soon as 2045[5]). And, at the point this "singularity" occurs, so commentators like Kevin Warwick[6] and Stephen Hawking[7] suggest, humanity will, effectively, have been "superseded" on the evolutionary ladder and be obliged to eke out its autumn days listening to "Industrial Metal" music and gardening; or, in some of Hollywood's even more dystopian dreams, cruelly subjugated (and/or exterminated) by "Terminator" machines.

In this paper, however, I will offer a few "critical reflections" on one of the central, albeit awkward, questions of AI: why is it that, seven decades since Alan Turing first deployed an "effective method" to play chess in 1948, we have seen enormous strides in engineering particular machines to do clever things—from driving a car to beating the best at Go—but almost no progress in getting machines to genuinely understand; to seamlessly apply knowledge from one domain into another—the so-called problem of "Artificial General Intelligence" (AGI); the skills that both Hollywood and the wider media really think of, and depict, as AI?

2. NEURAL COMPUTING

The earliest cybernetic work in the burgeoning field of "neural computing" lay in various attempts to understand, model, and emulate neurological function and learning in animal brains, the foundations of which were laid in 1943 by the neurophysiologist Warren McCulloch and the mathematician Walter Pitts (McCulloch and Pitts, 1943).

Neural Computing defines a mode of problem solving based on "learning from experience" as opposed to classical, syntactically specified, "algorithmic" methods; at its core is "*the study of networks of 'adaptable nodes' which, through a process of learning from task examples, store experiential knowledge and make it available for use*" (Aleksander and Morton, 1995). So construed, an "Artificial Neural Network" (ANN) is constructed merely by appropriately connecting a group of adaptable nodes ("artificial neurons").

- A *single layer neural network* only has one layer of adaptable nodes between the input vector, X and the output vector O, such that the output of each of the adaptable nodes defines one element of the network output vector O.
- A *multi-layer neural network* has one or more "hidden layers" of adaptable nodes between the input vector and the network output; in each of the network *hidden layers*, the outputs of the adaptable nodes connect to one or more inputs of the nodes in subsequent layers and in the network *output layer*, the output of each of the adaptable nodes defines one element of the network output vector O.
- A *recurrent neural network* is a network where the output of one or more nodes is fed-back to the input of other nodes in the architecture, such that the connections between nodes form a "directed graph along a temporal sequence," so enabling a recurrent network to exhibit "temporal dynamics," enabling a recurrent network to be sensitive to particular *sequences* of input vectors.

[1]Strong AI, a term coined by Searle (1980) in the "Chinese room argument" (CRA), entails that, "*... the computer is not merely a tool in the study of the mind; rather, the appropriately programmed computer really is a mind, in the sense that computers given the right programs can be literally said to understand and have other cognitive states*," which Searle contrasted with "Weak AI" wherein "*... the principal value of the computer in the study of the mind is that it gives us a very powerful tool.*" Weak AI focuses on epistemic issues relating to engineering a simulation of [human] intelligent behavior, whereas strong AI, in seeking to engineer a computational system with all the causal power of a mind, focuses on the ontological.

[2]See "[A]mplifiers for intelligence—devices that supplied with a little intelligence will emit a lot" (Ashby, 1956).

[3]See Moore's law: the observation that the number of transistors in a dense integrated circuit approximately doubles every 2 years.

[4]Conversely, as Francois Chollet, a senior engineer at Google and well-known scptic of the "Intelligence Explosion" scenario; trenchantly observed in 2017: "*The thing with recursive self-improvement in AI, is that if it were going to happen, it would already be happening. Auto-Machine Learning systems would come up with increasingly better Auto-Machine Learning systems, Genetic Programming would discover increasingly refined GP algorithms*" and yet, as Chollet insists, "*no human, nor any intelligent entity that we know of, has ever designed anything smarter than itself.*"

[5]Kurzweil (2005) "set the date for the Singularity—*representing a profound and disruptive transformation in human capability*—as 2045."

[6]In his 1997 book "March of the Machines", Warwick (1997) observed that there were already robots with the "*brain power of an insect*"; soon, or so he predicted, there would be robots with the "*brain power of a cat*," and soon after that there would be "*machines as intelligent as humans.*" When this happens, Warwick darkly forewarned, the science-fiction nightmare of a "Terminator" machine could quickly become reality because such robots will rapidly, and inevitably, become more intelligent and superior in their practical skills than the humans who designed and constructed them.

[7]In a television interview with Professor Stephen Hawking on December 2nd 2014, Rory Cellan-Jones asked how far engineers had come along the path toward creating Artificial Intelligence, to which Professor Hawking alarmingly replied, "*Once humans develop artificial intelligence it would take off on its own and redesign itself at an ever increasing rate. Humans, who are limited by slow biological evolution, couldn't compete, and would be superseded.*"

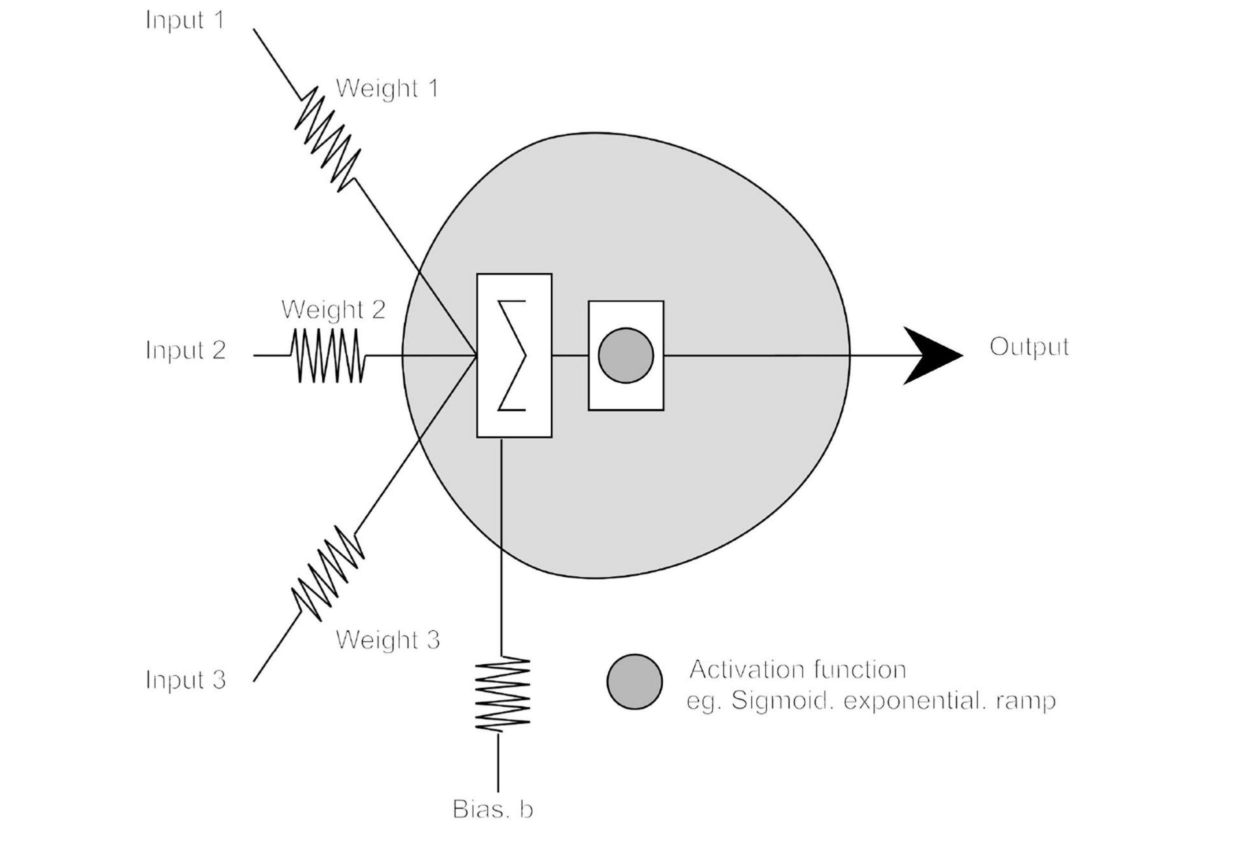

FIGURE 1 | The McCulloch–Pitts neuron model.

Since 1943 a variety of frameworks for the adaptable nodes have been proposed[8]; however, the most common, as deployed in many "deep" neural networks, remains grounded on the McCulloch/Pitts model.

2.1. The McCulloch/Pitts (MCP) Model

In order to describe how the basic processing elements of the brain might function, McCulloch and Pitts showed how simple electrical circuits, connecting groups of "linear threshold functions," could compute a variety of logical functions (McCulloch and Pitts, 1943). In their model, McCulloch and Pitts provided a first (albeit very simplified) mathematical account of the chemical processes that define neuronal operation and in so doing realized that the mathematics that describe the neuron operation exhibited exactly the same type of logic that Shannon deployed in describing the behavior of switching circuits: namely, the calculus of propositions.

McCulloch and Pitts (1943) realized (a) that neurons can receive positive or negative encouragement to fire, contingent upon the type of their "synaptic connections" (excitatory or inhibitory) and (b) that in firing the neuron has effectively performed a "computation"; once the effect of the excitatory/inhibitory synapses are taken into account, it is possible to *arithmetically* determine the net effect of incoming patterns of "signals" innervating each neuron.

In a simple McCulloch/Pitts (MCP) threshold model, adaptability comes from representing each synaptic junction by a variable (usually rational) valued weight W_i, indicating the degree to which the neuron should react to the $_ith$ particular input (see **Figure 1**). By convention, positive weights represent excitatory synapses and negative, inhibitory synapses; the neuron firing threshold being represented by a variable T. In modern use, T is usually clamped to zero and a threshold implemented using a variable "bias" weight, b; typically, a neuron firing[9] is represented by the value $+1$ and not firing by 0.

Activity at the i_{th} input to an n input neuron is represented by the symbol X_i and the effect of the i_{th} synapse by a weight W_i, hence the net effect of the i_{th} input on the i_{th} synapse on the MCP

[8] These include "spiking neurons" as widely used in computational neuroscience (Hodgkin and Huxley, 1952); "kernel functions" as deployed in "Radial Basis Function" networks (Broomhead and Lowe, 1988) and "Support Vector Machines" (Boser et al., 1992); "Gated MCP Cells," as deployed in LSTM networks (Hochreiter and Schmidhuber, 1997); "n-tuple" or "RAM" neurons, as used in "Weightless" neural network architectures (Bledsoe and Browning, 1959; Aleksander and Stonham, 1979), and "Stochastic Diffusion Processes" (Bishop, 1989) as deployed in the NESTOR multi-variate connectionist framework (Nasuto et al., 2009).

[9] "In psychology.. the fundamental relations are those of two valued logic" and McCulloch and Pitts recognized neuronal firing as equivalent to "representing" a proposition as *TRUE* or *FALSE* (McCulloch and Pitts, 1943).

cell is thus $X_i \times W_i$. Thus, the MCP cell is denoted as firing if:

$$\sum_{i}^{n} X_i \times W_i + b \geq 0 \quad (1)$$

In a subsequent generalization of the basic MCP neuron, cell output is defined by a further (typically non-linear) function of the weighted sum of its input, the neuron's *activation function*.

McCulloch and Pitts (1943) proved that if "synapse polarity" is chosen appropriately, any single pattern of input can be "recognized" by a suitable network of MCP neurons (i.e., any finite logical expression can be realized by a suitable network of McCulloch–Pitts neurons). In other words, the McCulloch–Pitts' result demonstrated that networks of artificial neurons could be mathematically specified, which would perform "computations" of immense complexity and power and in so doing, opened the door to a form of problem solving based on the design of appropriate neural network architectures and automatic (machine) "learning" of appropriate network parameters.

3. EMBEDDINGS IN EUCLIDEAN SPACE

The most commonly used framework for information representation and processing in artificial neural networks (via generalized McCulloch/Pitts neurons) is a subspace of Euclidean space. Supervised learning in this framework is equivalent to deriving appropriate transformations (learning appropriate mappings) from training data (problem exemplars; pairs of *Input* + "*Target Output*″ vectors). The majority of learning algorithms adjust neuron interconnection weights according to a specified "learning rule," the adjustment in a given time step being a function of a particular training example.

Weight updates are successively aggregated in this manner until the network reaches an equilibrium, at which point no further adjustments are made or, alternatively, learning stops before equilibrium to avoid "overfitting" the training data. On completion of these computations, knowledge about the training set is represented across a distribution of final weight values; thus, a trained network does not possess any internal representation of the (potentially complex) relationships *between* particular training exemplars.

Classical multi-layer neural networks are capable of discovering non-linear, continuous transformations between objects or events, but nevertheless they are restricted by operating on representations embedded in the linear, continuous structure of Euclidean space. It is, however, doubtful whether regression constitutes a satisfactory (or the most general) model of information processing in natural systems.

As Nasuto et al. (1998) observed, the world, and relationships between objects in it, is fundamentally non-linear; relationships between real-world objects (or events) are typically far too messy and complex for representations in Euclidean spaces—and smooth mappings between them—to be appropriate embeddings (e.g., entities and objects in the real-world are often fundamentally discrete or qualitatively vague in nature, in which case Euclidean space does not offer an appropriate embedding for their representation).

Furthermore, representing objects in a Euclidean space imposes a serious additional effect, because Euclidean vectors can be compared to each other by means of *metrics*; enabling data to be compared in spite of any real-life constraints (sensu stricto, metric rankings may be undefined for objects and relations of the real world). As Nasuto et al. (1998) highlight, it is not usually the case that all objects in the world can be equipped with a "natural ordering relation"; after all, what is the natural ordering of "banana" and "door"?

It thus follows that classical neural networks are best equipped only for tasks in which they process numerical data whose relationships can be reflected by Euclidean distance. In other words, classical connectionism can be reasonably well-applied to the same category of problems, which could be dealt with by various regression methods from statistics; as Francois Chollet[10], in reflecting on the limitations of deep learning, recently remarked:

> "[a] deep learning model is 'just' a chain of simple, continuous geometric transformations mapping one vector space into another. All it can do is map one data manifold X into another manifold Y, assuming the existence of a learnable continuous transform from X to Y, and the availability of a dense sampling of X: Y to use as training data. So even though a deep learning model can be interpreted as a kind of program, inversely most programs cannot be expressed as deep learning models-for most tasks, either there exists no corresponding practically-sized deep neural network that solves the task, or even if there exists one, it may not be learnable … most of the programs that one may wish to learn cannot be expressed as a continuous geometric morphing of a data manifold" (Chollet, 2018).

Over the last decade, however, ANN technology has developed beyond performing "simple function approximation" (cf. Multi-Layer Perceptrons) and deep [discriminative[11]] classification (cf. Deep Convolutional Networks), to include new, *Generative* architectures[12] where—*because they can learn to generate any distribution of data*—the variety of potential use cases is huge (e.g., generative networks can be taught to create novel outputs similar to real-world exemplars across any modality: images, music, speech, prose, etc.).

3.1. Autoencoders, Variational Autoencoders, and Generative Adversarial Networks

On the right hand side of **Figure 2**, we see the output of a neural system, engineered by Terence Broad while studying for an MSc at Goldsmiths. Broad used a "complex, deep auto-encoder neural network" to process Blade Runner—a well-known sci-fi film that riffs on the notion of what is human and

[10]Chollet is a senior software engineer at Google, who—as the primary author and maintainer of Keras, the Python open source neural network interface designed to facilitate fast experimentation with Deep Neural Networks—is familiar with the problem-solving capabilities of deep learning systems.

[11]A discriminative architecture—or discriminative classifier without a model—can be used to "discriminate" the value of the target variable Y, given an observation x.

[12]A generative architecture can be used to "generate" random instances, either of an observation and target (x, y), or of an observation x given a target value y.

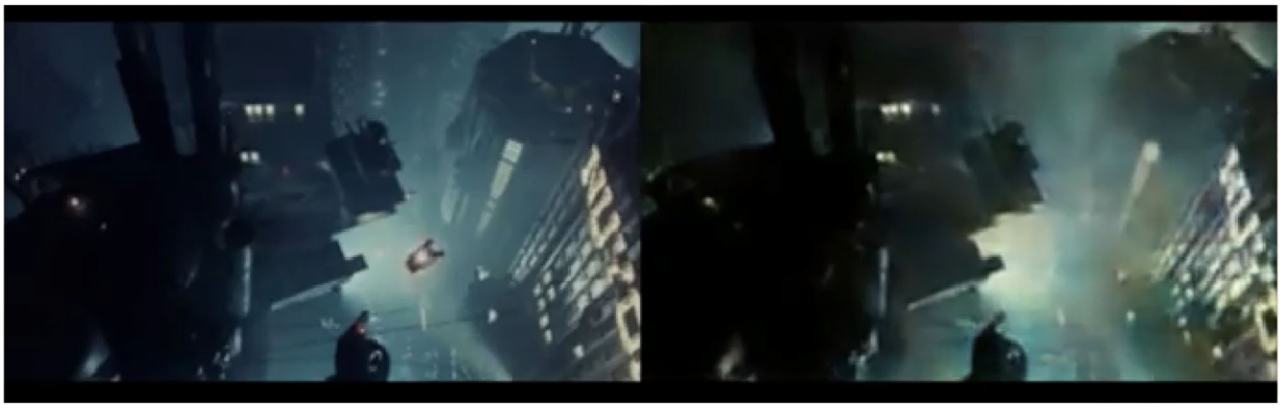

FIGURE 2 | Terrence Broad's Auto-encoding network "dreams" of Bladerunner (from Broad, 2016).

what is machine—building up its own "internal representations" of that film and then re-rendering these to produce an output movie that is surprisingly similar to the original (shown on the left).

In Broad's dissertation (Broad, 2016), a "Generative Autoencoder Network" reduced each frame of Ridley Scott's Blade Runner to 200 "latent variables" (hidden representations), then invoked a "decoder network" to reconstruct each frame just using those numbers. The result is eerily suggestive of an Android's dream; the network, working without human instruction, was able to capture the most important elements of each frame so well that when its reconstruction of a clip from the Blade Runner movie was posted to Vimeo, it triggered a "Copyright Takedown Notice" from Warner Brothers.

To understand if Generative Architectures are subject to the Euclidean constraints identified above for classical neural paradigms, it is necessary to trace their evolution from the basic Autoencoder Network, through Variational Autoencoders to Generative Adversarial Networks.

3.1.1. Autoencoder Networks

"Autoencoder Networks" (Kramer, 1991) create a latent (or hidden), typically much compressed, representation of their input data. When Autoencoders are paired with a decoder network, the system can reverse this process and reconstruct the input data that generates a particular latent representation. In operation, the Autoencoder Network is given a data input x, which it maps to a latent representation z, from which the decoder network reconstructs the data input x' (typically, the cost function used to train the network is defined as the mean squared error between the input x and the reconstruction x'). Historically, Autoencoders have been used for "feature learning" and "reducing the dimensionality of data" (Hinton and Salakhutdinov, 2006), but more recent variants (described below) have been powerfully deployed to learn "Generative Models" of data.

3.1.2. Variational Autoencoder Networks

In taking a "variational Bayesian" approach to learning the hidden representation, "Variational Autoencoder Networks" (Kingma and Welling, 2013) add an additional constraint, placing a strict assumption on the distribution of the latent variables. Variational Autoencoder Networks are capable of both compressing data instances (like an Autoencoder) and generating new data instances.

3.1.3. Generative Adversarial Networks

Generative Adversarial Networks (Goodfellow et al., 2014) deploy two "adversary" neural networks: one, the Generator, synthesizes new data instances, while the other, the Discriminator, rates each instance as how likely it is to belong to the training dataset. Colloquially, the Generator takes the role of a "counterfeiter" and the Discriminator the role of "the police," in a complex and evolving game of cat and mouse, wherein the counterfeiter is evolving to produce better and better counterfeit money while the police are getting better and better at detecting it. This game goes on until, at convergence, both networks have become very good at their tasks; Yann LeCun, Facebook's AI Director of Research, recently claimed them to be "*the most interesting idea in the last ten years in Machine Learning*"[13].

Nonetheless, as Goodfellow emphasizes (Goodfellow et al., 2014), the generative modeling framework is most straightforwardly realized using "multilayer perceptron models." Hence, although the functionally of generative architectures moves beyond the simple function-approximation and discriminative-classification abilities of classical multi-layer perceptrons, at heart, in common with all neural networks that learn, and operate on, functions embedded in Euclidean space[14], they remain subject to the constraints of Euclidean embeddings highlighted above.

[13]Quora July 28, 2016 (https://www.quora.com/session/Yann-LeCun/1).

[14]Including neural networks constructed using alternative "adaptable node" frameworks (e.g., those highlighted in footnote [8]), where these operate on data embeddings in Euclidean space.