Artificial Intelligence Engines: A Tutorial Introduction to the Mathematics of Deep Learning

Book · April 2019

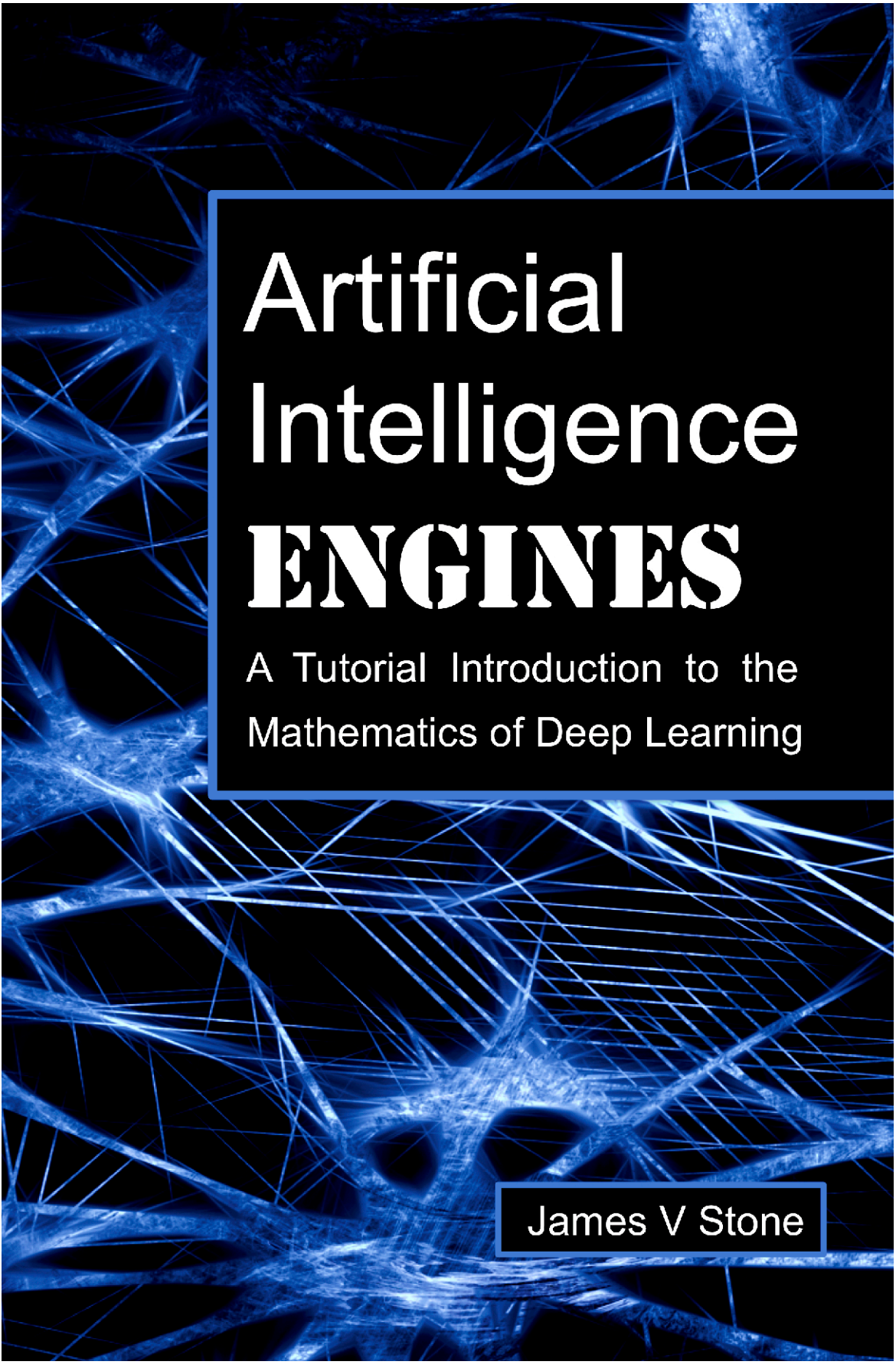
CITATIONS
4

READS
10,056

1 author:

James V Stone
The University of Sheffield
81 PUBLICATIONS **2,669** CITATIONS

SEE PROFILE


Artificial
Intelligence
ENGINES
A Tutorial Introduction to the
Mathematics of Deep Learning
James V Stone

Artificial Intelligence Engines

A Tutorial Introduction to the Mathematics of Deep Learning

James V Stone

Title: Artificial Intelligence Engines
Author: James V Stone

First Edition, 2019.
Typeset in LaTeX$\partial 2_\varepsilon$.
First printing.

ISBN 9780956372819

Cover background by Okan Caliska, reproduced with permission.

For Teleri, my bright star

Contents

List of Pseudocode Examples

Each example below refers to a text box that summarises a particular method or neural network.

Online Code Examples

The Python and MatLab code examples below can be obtained from the online GitHub repository at
`https://github.com/jgvfwstone/ArtificialIntelligenceEngines`
Please see the file README.txt in that repository.

The computer code has been collated from various sources (with permission). It is intended to provide small scale transparent examples, rather than an exercise in how to program artificial neural networks.

The examples listed below are written in Python, usually with the PyTorch framework. Additional examples are duplicated using MatLab.

Linear network

Perceptron

Backprop network

Hopfield net

Boltzmann machine

Restricted Boltzmann machine

Variational autoencoder

Convolutional neural network

Reinforcement learning

Preface

This book is intended to provide an account of deep neural networks that is both informal and rigorous. Each neural network learning algorithm is introduced informally with diagrams, before a mathematical account is provided. In order to cement the reader's understanding, each algorithm is accompanied by a step-by-step summary written in pseudocode. Thus, each algorithm is approached from several convergent directions, which should ensure that the diligent reader gains a thorough understanding of the inner workings of deep learning artificial neural networks. Additionally, technical terms are described in a comprehensive glossary, and (for the complete novice) a tutorial appendix on vectors and matrices is provided.

Unlike most books on deep learning, this is not a 'user manual' for any particular software package. Such books often place high demands on the novice, who has to learn the conceptual infrastructure of neural network algorithms, whilst simultaneously learning the minutiae of how to operate an unfamiliar software package. Instead, this book concentrates on key concepts and algorithms.

Having said that, readers familiar with programming can benefit from running working examples of neural networks, which are available at the website associated with this book. Simple examples were written by the author, and these are intended to be easy to understand rather than efficient to run. More complex examples have been borrowed (with permission) from around the internet, but mainly from the PyTorch repository. A list of the online code examples that accompany this book is given opposite.

Who Should Read This Book? The material in this book should be accessible to anyone with an understanding of basic calculus. The tutorial style adopted ensures that any reader prepared to put in the effort will be amply rewarded with a solid grasp of the fundamentals of deep learning networks.

The Author. My education in artificial neural networks began with a lecture by Geoff Hinton on Boltzmann machines in 1984 at Sussex University, where I was studying for an MSc in Knowledge Based Systems. I was so inspired by this lecture that I implemented Hinton and Sejnowski's Boltzmann machine for my MSc project. Later, I enjoyed extended visits to Hinton's laboratory in Toronto and to Sejnowski's laboratory in San Diego, funded by a Wellcome Trust Mathematical Biology Fellowship. I have published research papers on human memory[30;125], temporal backprop[116], backprop[71], evolution[120], optimisation[115;117], independent component analysis[119], and using neural networks to test principles of brain function[118;124].

Acknowledgements. Thanks to Steve Snow for discussions on variational autoencoders and the physics of Boltzmann machines. For debates about neural networks over many years, I am indebted to my friend Raymond Lister. For helping with Python frameworks, thanks to Royston Sellman. For reading one or more chapters, I am very grateful to Olivier Codol, Peter Dunne, Lancelot Da Costa, Stephen Eglen, George Farmer, Charles Fox, Nikki Hunkin, Raymond Lister, Gonzalo de Polavieja, and Paul Warren. Special thanks to Eleni Vasilaki and Luca Manneschi for providing guidance on reinforcement learning. For allowing me to use their computer code, I'd like to thank Jose Antonio Martin H (reinforcement learning) and Peter Dunne (for his version of Hinton's RBM code). Finally, thanks to Alice Yew for meticulous copy-editing and proofreading.

Corrections. Please email corrections to j.v.stone@sheffield.ac.uk.
A list of corrections can be found at:
`http://jim-stone.staff.shef.ac.uk/AIEngines`

Once the machine thinking method has started, it would not take long to outstrip our feeble powers.
A Turing, 1951.

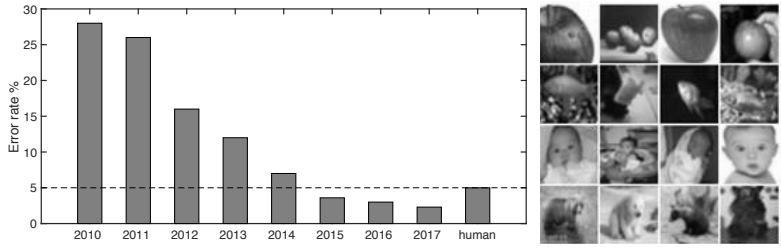
Chapter 1

Artificial Neural Networks

If you want to understand a really complicated device, like a brain, you should build one.
G Hinton, 2018.

1.1. Introduction

Deep neural networks are the computational engines that power modern artificial intelligence. Behind every headline regarding the latest achievements of artificial intelligence lies a deep neural network, whether this involves diagnosing cancer, recognising objects (Figure 1.1), learning to ride a bicycle[90], or learning to fly[28;93] (Figure 1.2). In essence, deep neural networks rely on a sophisticated form of pattern recognition, but the secret of their success is that they are capable of *learning*.



(a) (b)

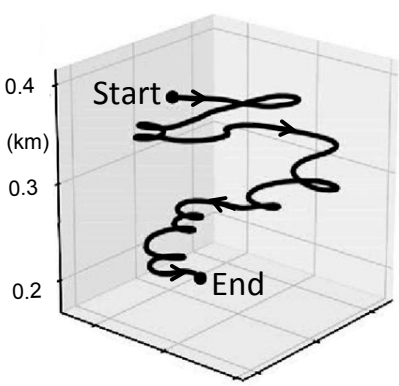
Figure 1.1. Classifying images. (a) The percentage error on the annual ILSVRC image classification competition has fallen dramatically since 2010. (b) Example images. The latest competition involves classifying about 1.5 million images into 1,000 object classes. Reproduced with permission from Wu and Gu (2015).

A striking indication of the dramatic progress made over recent years can be seen in Figure 1.1, which shows performance on an image classification task. Note that the performance of deep neural networks surpassed human performance in 2015.

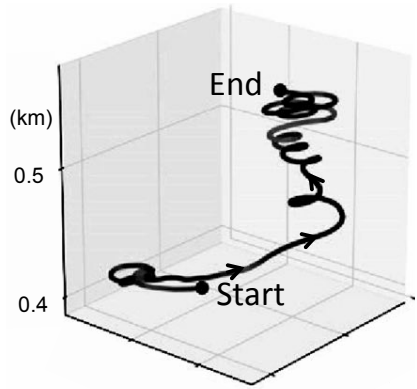
1.2. What is an Artificial Neural Network?

An artificial neural network is a set of interconnected model neurons or *units*, as shown in Figure 1.3. For brevity, we will use the term *neural network* to refer to artificial neural networks. By analogy to the brain, the most common forms of neural networks receive inputs via an array of sensory units. With repeated presentations, a neural network

(a)



(b)



(c)

Figure 1.2. Learning to soar using reinforcement learning. (a) Glider used for learning. (b) Before learning, flight is disorganised, and glider descends. (c) After learning, glider ascends to 0.6 km. Note the different scales on the vertical axes in (b) and (c). Reproduced with permission: (a) from Guilliard et al. (2018); (b,c) from Reddy et al. (2016).

can learn to recognise those inputs so as to classify them or to perform some action. For example, if the input to a neural network is an image of a teapot then the output should indicate that the image contains a teapot. However, this simple description hides the many complexities that make object recognition extraordinarily difficult. Indeed, the fact that we humans are so good at object recognition makes it hard to appreciate just how difficult it is for a neural network to recognise objects. A teapot can vary in colour and shape, and it can appear at many different orientations and sizes, as in Figure 1.4. These variations do not usually fool humans, but they can cause major problems for neural networks.

Let's assume that we have wired up a neural network so that its input is an image, and that we want its output to indicate which object is in the image. Specifically, we wire up the network so that each pixel in the image acts as input to one *input unit*, and we connect all of these input units to a single *output unit*. The output or *state* of the output unit is determined by the image through the strengths or *weights* of the connections from the different input units to the output unit. In essence, the bigger the weight, the more a given input unit affects the state of the output unit. To make things really simple, we assume that the object is either a teapot or a cup, and that the state of the output unit should be 1 when the image contains a teapot and 0 when it contains a cup.

In order to get a feel for the magnitude of the problem confronting a neural network, consider a square (black and white) image that is 1,000 × 1,000 pixels. If each pixel can adopt 256 grey-levels then the

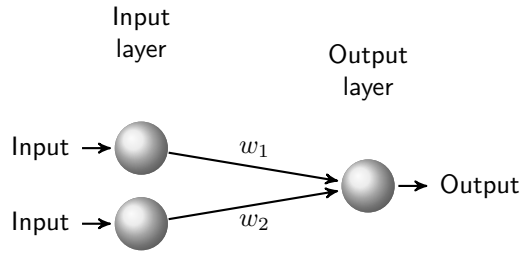


Figure 1.3. A neural network with two input units and one output unit. The connections from the input units to the output unit have weights w_1 and w_2.

number of possible images is $256^{1,000,000}$, which is much more than the number of atoms in the universe (about 10^{87}). Of course, most of these images look like random noise, and only a tiny fraction depict anything approaching a natural scene — and only a fraction of those depict a teapot. The point is that if a network is to discriminate between teapots and non-teapots then it implicitly must learn to assign a probability to each of the $256^{1,000,000}$ possible images, which is the probability that each image contains a teapot.

The neural network has to learn to recognise objects in its input image by adjusting the connection weights between units (here, between input and output units). The process of adjusting weights is implemented by a learning *algorithm* (the word *algorithm* just means a definite sequence of logical steps that yields a specified outcome).

Now, suppose the input image contains a teapot but the output unit state is 0 (indicating that the image depicts a cup). This *classification error* occurs because one or more weights have the wrong values. However, it is far from obvious which weights are wrong and how they should be adjusted to correct the error. This general problem is called the *credit assignment problem*, but it should really be called the blame assignment problem, or even the whodunnit problem.

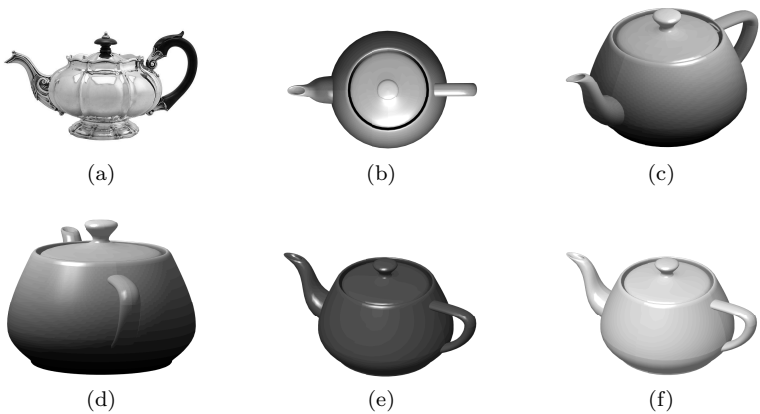


Figure 1.4. Teapots come in various shapes (a vs b), and the single teapot in (b)–(f) can appear with many different orientations, sizes, and colours (represented as shades of grey here).

In practice, knowing whodunnit, and what to do with whoever-did-do-it, represents the crux of the problem to be solved by neural network learning algorithms. In order to learn, artificial (and biological) neural networks must solve this credit/blame assignment problem.

Most solutions to this problem have their roots in an idea proposed by Donald Hebb[34] in 1949, which is known as *Hebb's postulate*:

> *When an axon of cell A is near enough to excite cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.*

1.3. The Origins of Neural Networks

The development of modern neural networks (see Figure 1.5) occurred in a series of step-wise dramatic improvements, interspersed with periods of stasis (often called *neural network winters*). As early as 1943, before commercial computers existed, McCulloch and Pitts[77] began to explore how small networks of artificial neurons could mimic brain-like processes. The cross-disciplinary nature of this early research is evident from the fact that McCulloch and Pitts also contributed to a classic paper on the neurophysiology of vision (What the frog's eye tells the frog's brain[68], 1959), which was published not in a biology journal, but in the *Proceedings of the Institute of Radio Engineers*.

In subsequent years, the increasing availability of computers allowed neural networks to be tested on simple pattern recognition tasks. But progress was slow, partly because most research funding was allocated to more conventional approaches that did not attempt to mimic the neural architecture of the brain, and partly because artificial neural network learning algorithms were limited.

A major landmark in the history of neural networks was Frank Rosenblatt's *perceptron* (1958), which was explicitly modelled on the neuronal structures in the brain. The *perceptron learning algorithm* allowed a perceptron to learn to associate inputs with outputs in a manner apparently similar to learning in humans. Specifically, the perceptron could learn an association between a simple input 'image'

and a desired output, where this output indicated whether or not the object was present in the image.

Perceptrons, and neural networks in general, share three key properties with human memory. First, unlike conventional computer memory, neural network memory is *content addressable*, which means that recall is triggered by an image or a sound. In contrast, computer memory can be accessed only if the specific location (address) of the required information is known. Second, a common side-effect of content addressable memory is that, given a learned association between an input image and an output, recall can be triggered by an input image that is similar (but not identical) to the original input of a particular input/output association. This ability to *generalise* beyond learned associations is a critical human-like property of artificial neural networks. Third, if a single weight or unit is destroyed, this does not remove a particular learned association; instead, it degrades all associations to some extent. This *graceful degradation* is thought to resemble human memory.

Despite these human-like qualities, the perceptron was dealt a severe blow in 1969, when Minsky and Papert[79] famously proved that it could not learn associations unless they were of a particularly simple kind (i.e. *linearly separable*, as described in Chapter 2). This marked the beginning of the first neural network winter, during which neural network research was undertaken by only a handful of scientists.

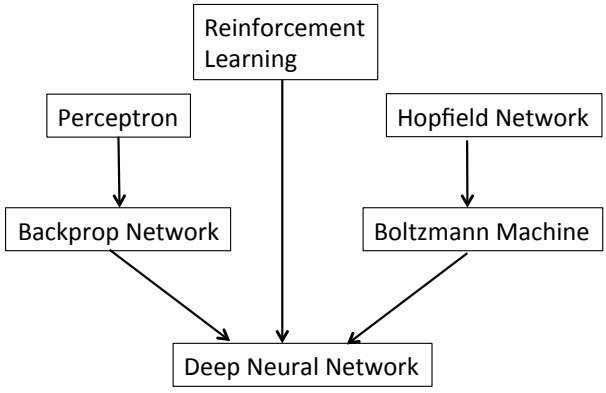


Figure 1.5. A simplified taxonomy of artificial neural networks.

The modern era of neural networks began in 1982 with the *Hopfield net*[47]. Although Hopfield nets are not practically very useful, Hopfield introduced a theoretical framework based on *statistical mechanics*, which laid the foundations for Ackley, Hinton, and Sejnowki's *Boltzmann machine*[1] in 1985. Unlike a Hopfield net, in which the *states* of all units are specified by the associations being learned, a Boltzmann machine has a reservoir of *hidden units*, which can be used to learn complex associations. The Boltzmann machine is important because it facilitated a conceptual shift away from the idea of a neural network as a passive associative machine towards the view of a neural network as a *generative model*. The only problem is that Boltzmann machines learn at a rate that is best described as glacial. But on a practical level, the Boltzmann machine demonstrated that neural networks could learn to solve complex toy (i.e. small-scale) problems, which suggested that they could learn to solve almost any problem (at least in principle, and at least eventually).

The impetus supplied by the Boltzmann machine gave rise to a more tractable method devised in 1986 by Rumelhart, Hinton, and Williams, the *backpropagation learning algorithm*[97]. A backpropagation network consists of three layers of units: an *input layer*, which is connected with connection weights to a hidden layer, which in turn is connected to an *output layer*, as shown in Figure 1.6. The backpropagation algorithm is important because it demonstrated the potential of neural networks to learn sophisticated tasks in a human-like manner. Crucially, for the first time, a backpropagation neural network called *NetTalk* learned to 'speak', inasmuch as it translated text to *phonemes* (the basic elements of speech), which a voice synthesizer then used to produce speech.

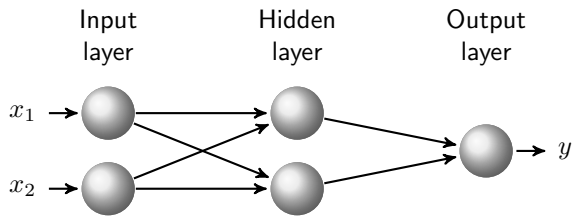


Figure 1.6. A neural network with two input units, two hidden units, and one output unit.

In parallel with the evolution of neural networks, *reinforcement learning* was developed throughout the 1980s and 1990s, principally by Sutton and Barto (2018). Reinforcement learning is an inspired fusion of game playing by computers, as developed by Shannon (1950) and Samuel (1959), optimal control theory, and stimulus–response experiments in psychology. Early results showed that hard, albeit small-scale, problems such as balancing a pole can be solved using feedback in the form of simple reward signals (Michie and Chambers, 1968; Barto, Sutton, and Anderson, 1983). More recently, reinforcement learning has been combined with deep learning to produce impressive skill acquisition, such as in the case of a glider that learns to gain height on thermals[28] (see Figure 1.2).

1.4. From Backprop to Deep Learning

In principle, a backprop network with just two hidden layers of units can associate pretty much any inputs with any outputs[12], which means that it should be able to perform most tasks. However, getting a backprop to *learn* the tasks that it should be able to perform in theory is problematic. A plausible solution is to add more units to each layer and more layers of hidden units, because this should improve learning (at least in theory). In practice, it was found that conventional backprop networks struggled to learn if they had deep learning architectures like that shown in Figure 1.7.

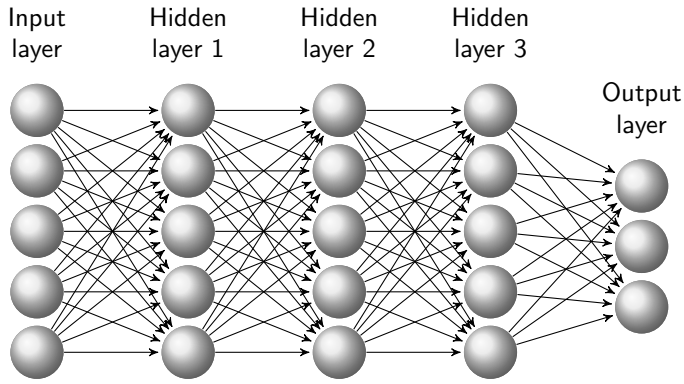


Figure 1.7. A deep network with three hidden layers.

With the benefit of hindsight, the field of artificial intelligence stems from the research originally done on Hopfield nets, Boltzmann machines, the backprop algorithm, and reinforcement learning. However, the evolution of backprop networks into deep learning networks had to wait for three related developments. To quote one of the inventors of backprop, the reason for the revolution in neural networks is because:

> *In simple terms, computers became millions of times faster, data sets got thousands of times bigger, and researchers discovered many technical refinements that made neural nets easier to train.*
> G Hinton, 2018.

1.5. An Overview of Chapters

In order to appreciate modern deep learning networks, some familiarity with their historical antecedents is required. Accordingly, Chapter 2 provides a tutorial on the simplest neural network, a *linear associative network*. This embodies many of the essential ingredients (e.g. *gradient descent*) necessary for understanding more sophisticated neural networks. In Chapter 3, the role of the *perceptron* in exposing the limitations of linear neural networks is described.

In Chapter 4, the *backpropagation* or *backprop* algorithm is introduced and demonstrated on the simplest nonlinear problem (i.e. the exclusive OR problem). The application of *recurrent backprop neural networks* to the problem of recognising sequences (e.g. speech) is also explored. The properties of backprop are examined in the context of the *cross-entropy error function* and in relation to problems of *overfitting*, *generalisation*, and *vanishing gradients*.

To understand the stochastic nature of deep learning neural networks, Chapter 5 examines *Hopfield nets*, which have their roots in statistical mechanics. In Chapter 6, the limitations of Hopfield nets are addressed in the form of *Boltzmann machines*, which represent the first *generative models*. The glacial learning rates of Boltzmann machines are overcome by *restricted Boltzmann machines* in Chapter 7. A recent

innovation in the form of *variational autoencoders* promises a further step-change in generative models, as described in Chapter 8.

The use of convolution in backprop networks in the early 1990s foreshadowed the rise of *deep convolutional networks*, *ladder networks*, *denoising autoencoders*, *generative adversarial networks*, *temporal deep networks*, and *capsule networks*, which are described in Chapter 9.

In Chapter 10, the *reinforcement learning algorithm* is introduced and demonstrated on examples such as pole balancing. Ingenious variants of backprop were devised between 2000 and 2016, which, when combined with reinforcement learning, allowed deep networks to learn increasingly complex tasks. A particularly striking result of these developments was a deep network called AlphaGo[108], which learned to play the game of Go so well that it beat the world's best human players in 2016.

Finally, the prospect of an artificial intelligence revolution is discussed in Chapter 11, where we consider the two extremes of naive optimism and cynical pessimism that inevitably accompany any putative revolution.

References

[1] Ackley, DH, Hinton, GE, and Sejnowski, TJ. A learning algorithm for Boltzmann machines. *Cog. Sci.*, 9:147–169, 1985.

[2] Alemi, AA, Fischer, I, Dillon, JV, and Murphy, K. Deep variational information bottleneck. *arXiv e-prints*, 2016. arXiv:1612.00410.

[3] Bai, S, Kolter, Z, and Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv e-prints*, 2018. arXiv:1803.01271.

[4] Barto, AG, Sutton, RS, and Anderson, CW. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Sys. Man Cyb.*, 13(5):834–846, 1983.

[5] Bishop, CM. *Neural Networks for Pattern Recognition*. Oxford University Press, 1996.

[6] Blum, A and Rivest, RL. Training a 3-node neural network is NP-complete. In *NIPS*, pages 494–501, 1989.

[7] Bruineberg, J, Rietveld, E, Parr, T, van Maanen, L, and Friston, KJ. Free-energy minimization in joint agent-environment systems: A niche construction perspective. *J. Theor. Biol.*, 455:161–178, 2018.

[8] Burda, Y, Grosse, R, and Salakhutdinov, R. Importance weighted autoencoders. *arXiv e-prints*, 2015. arXiv:1509.00519.

[9] Burgess, CP, Higgins, I, Pal, A, Matthey, L, Watters, N, Desjardins, G, and Lerchner, A. Understanding disentangling in β-VAE. *arXiv e-prints*, 2018. arXiv:1804.03599.

[10] Cho, K, van Merrienboer, B, Gulcehre, C, Bahdanau, D, Bougares, F, Schwenk, H, and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv e-prints*, 2014. arXiv:1406.1078.

[11] Choromanska, A, Henaff, M, Mathieu, M, Ben Arous, G, and LeCun, Y. The loss surfaces of multilayer networks. *arXiv e-prints*, 2014. arXiv:1412.0233.

[12] Cybenko, G. Continuous valued neural networks with two hidden layers are sufficient. Technical report, Department of Computer Science, Tufts University, Medford, Massachusetts, 1988.

[13] Cybenko, G. Approximation by superposition of a sigmoidal function. *Maths Contr. Sig. Sys.*, 2:303–314, 1989.

[14] Dayan, P and Watkins, CJCH. Q-Learning. *Machine Learning*, 8(3):279–292, 1992.

[15] Doersch, C. Tutorial on variational autoencoders. *arXiv e-prints*, 2016. arXiv:1606.05908.

[16] Doya, K. Reinforcement learning in continuous time and space. *Neural Comptutation*, 12(1):219–245, 2000.

[17] Du, SS, Lee, JD, Li, H, Wang, L, and Zhai, X. Gradient descent finds global minima of deep neural networks. *arXiv e-prints*, 2018. arXiv:1811.03804.

[18] Elbayad, M, Besacier, L, and Verbeek, J. Pervasive attention: 2D convolutional neural networks for sequence-to-sequence prediction. *arXiv e-prints*, 2018. arXiv:1808.03867.

[19] Elman, JL. Finding structure in time. *Cog. Sci.*, 14(2):179–211, 1990.

[20] Eysenbach, B, Gupta, A, Ibarz, J, and Levine, S. Diversity is all you need: Learning diverse skills without a reward function. *arXiv e-prints*, 2018. arXiv:1802.06070.

[21] Gallagher, M, Downs, T, and Wood, I. Ultrametric structure in autoencoder error surfaces. In *ICANN 98*, pages 177–182. Springer, 1998.

[22] Geman, S and Geman, D. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *J. Appl. Statist.*, 20:25–62, 1993.

[23] Gers, FA, Schmidhuber, J, and Cummins, F. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 2000.

[24] Glorot, X and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proc. 13th Int. Conf. AI and Statistics*, pages 249–256, 2010.

[25] Goodfellow, IJ, Pouget-Abadie, J, Mirza, M, Xu, B, Warde-Farley, D, Ozair, S, Courville, A, and Bengio, Y. Generative adversarial networks. *arXiv e-prints*, 2014. arXiv:1406.2661.

[26] Gorman, RP and Sejnowski, TJ. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, 1(1):75–89, 1988.

[27] Greener, JG, Moffat, L, and Jones, DT. Design of metalloproteins and novel protein folds using variational autoencoders. *Scientific Reports*, 8(1):16189, 2018.

[28] Guilliard, I, Rogahn, R, Piavis, J, and Kolobov, A. Autonomous thermalling as a partially observable Markov decision process. *arXiv e-prints*, 2018. arXiv:1805.09875.

[29] Haarnoja, T, Zhou, A, Ha, S, Tan, J, Tucker, G, and Levine, S. Learning to walk via deep reinforcement learning. *arXiv e-prints*, 2018. arXiv:1812.11103.

[30] Harvey, I and Stone, JV. Unicycling helps your French: Spontaneous recovery of associations by learning unrelated tasks. *Neural Computation*, 8:697–704, 1996.

[31] Hayakawa, H, Inui, T, and Kawato, M. Computational theory and neural network model of perceiving shape from shading in monocular

depth perception. In *IJCNN-91-Seattle*, pages 649–654, 1991.

[32] He, K, Zhang, X, Ren, S, and Sun, J. Deep residual learning for image recognition. *arXiv e-prints*, 2015. arXiv:1512.03385.

[33] He, K, Zhang, X, Ren, S, and Sun, J. Identity mappings in deep residual networks. In *ECCV 2016*, pages 630–645. Springer, 2016.

[34] Hebb, DO. *The Organization of Behavior: A Neuropsychological Theory*. Wiley, New York, 1949.

[35] Helmbold, D and Long, P. Surprising properties of dropout in deep networks. *Journal of Machine Learning Research*, 18(1):7284–7311, 2017.

[36] Hertz, J, Krogh, A, and Palmer, RG. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[37] Higgins, I, Matthey, L, Glorot, X, Pal, A, Uria, B, Blundell, C, Mohamed, S, and Lerchner, A. Early visual concept learning with unsupervised deep learning. *arXiv e-prints*, 2016. arXiv:1606.05579.

[38] Hinton, G. Deep learning: A technology with the potential to transform health care. *JAMA*, 320(11):1101–1102, 2018.

[39] Hinton, GE. A parallel computation that assigns canonical object-based frames of reference. *Proc. 7th Int. Joint Conf. AI*, pages 683–685, 1981.

[40] Hinton, GE. Shape representation in parallel systems. In *Proc. 7th Int. Joint Conf. AI*, pages 1088–1096, 1981.

[41] Hinton, GE. A practical guide to training restricted Boltzmann machines. In *Neural Networks: Tricks of the Trade*, pages 599–619. Springer, 2012.

[42] Hinton, GE, Osindero, S, and Teh, Y. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[43] Hinton, GE and Salakhutdinov, RR. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[44] Hinton, GE, Sejnowski, TJ, and Ackley, DH. Boltzmann machines: Constraint satisfaction networks that learn. Technical report, Department of Computer Science, Carnegie-Mellon University, 1984.

[45] Hinton, GH, Osindero, S, and Teh, YW. Is the early visual system optimised to be energy efficient? *Neural Computation*, 18(7):1527–1554, 2006.

[46] Hochreiter, S and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[47] Hopfield, JJ. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA*, 79(8):2554–2558, 1982.

[48] Hopfield, JJ. Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. Nat. Acad. Sci. USA*, 81:3088–3092, 1984.

[49] Houthooft, R, Chen, X, Duan, Y, Schulman, J, De Turck, F, and Abbeel, P. VIME: Variational information maximizing exploration. *arXiv e-prints*, 2016. arXiv:1605.09674.

[50] Hsu, W, Zhang, Y, and Glass, J. Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation. *arXiv e-prints*, 2017. arXiv:1707.06265.

[51] Huang, G, Liu, Z, Maaten, L, and Weinberger, KQ. Densely connected convolutional networks. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[52] Hwangbo, J and et al. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019.

[53] Ioffe, S and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[54] Jordan, MI. Serial order: A parallel distributed approach. Technical report, Institute for Cognitive science, University of California, San Diego, 1986. ICS Report 8604.

[55] Karras, T, Aila, T, Laine, S, and Lehtinen, J. Progressive growing of GANs for improved quality, stability, and variation. *arXiv e-prints*, 2017. arXiv:1710.10196.

[56] Kavasidis, I, Palazzo, S, Spampinato, C, Giordano, D, and Shah, M. Brain2Image: Converting brain signals into images. In *Proc. 25th ACM Int. Conf. Multimedia*, pages 1809–1817. ACM, 2017.

[57] Kim, Y, Wiseman, S, and Rush, AM. A tutorial on deep latent variable models of natural language. *arXiv e-prints*, 2015. arXiv:1812.06834.

[58] Kingma, DP. *Variational Inference & Deep Learning*. PhD thesis, University of Amsterdam, 2017.

[59] Kingma, DP, Mohamed, S, Rezende, DJ, and Welling, M. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

[60] Kingma, DP and Welling, M. Auto-encoding variational Bayes. *arXiv e-prints*, 2013. arXiv:1312.6114.

[61] Kohonen, T. Correlation matrix memories. *IEEE Trans. Computers*, 100(4):353–359, 1972.

[62] Krizhevsky, A, Sutskever, I, and Hinton, GE. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[63] Larochelle, H, Bengio, Y, Louradour, J, and Lamblin, P. Exploring strategies for training deep neural networks. *Journal of Machine Learning Research*, 10(Jan):1–40, 2009.

[64] Larsen, A, Sønderby, S, Larochelle, H, and Winther, O. Autoencoding beyond pixels using a learned similarity metric. *arXiv e-prints*, 2015. arXiv:1512.09300.

[65] Le Roux, N and Bengio, Y. Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649, 2008.

[66] LeCun, Y, Boser, B, Denker, JS, Henderson, RE, Hubbard, W, and Jackel, LD. Backpropagation applied to handwritten ZIP code recognition. *Neural Computation*, 1:541–551, 1989.

[67] LeCun, Y, Bottou, L, Bengio, Y, and Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[68] Lettvin, JV, Maturana, HR, McCulloch, WS, and Pitts, WH. What the frog's eye tells the frog's brain. *Proceedings of the Institute of Radio Engineers*, pages 1940–1951, 1959.

[69] Lim, J, Ryu, S, Kim, JW, and Kim, WY. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *arXiv e-prints*, 2018. arXiv:1806.05805.

[70] Lister, R. Annealing networks and fractal landscapes. In *IEEE Int. Conf. Neural Networks (San Francisco)*, pages 257–262, 1993.

[71] Lister, R and Stone, JV. An empirical study of the time complexity of various error functions with conjugate gradient back propagation. *IEEE Int. Conf. Artificial Neural Networks (ICNN'95, Perth, Australia)*, 1995.

[72] Longuet-Higgins, HC. The non-local storage of temporal information. *Proc. R. Soc. Lond. B*, 171(1024):327–334, 1968.

[73] Longuet-Higgins, HC, Willshaw, DJ, and Buneman, OP. Theories of associative recall. *Quarterly Reviews of Biophysics*, 3(2):223–244, 1970.

[74] Maaløe, L, Fraccaro, M, Liévin, V, and Winther, O. BIVA: A very deep hierarchy of latent variables for generative modeling. *arXiv e-prints*, 2019. arXiv:1902.02102.

[75] MacKay, DJC. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[76] Marcus, G. Deep learning: A critical appraisal. *arXiv e-prints*, 2018. arXiv:1801.00631.

[77] McCulloch, WS and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophysics*, 5:115–133, 1943.

[78] Michie, D and Chambers, RA. BOXES: An experiment in adaptive control. *Machine Intelligence*, 2(2):137–152, 1968.

[79] Minsky, M and Papert, S. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[80] Mnih, V et al. Playing Atari with deep reinforcement learning. *arXiv e-prints*, 2013. arXiv:1312.5602.

[81] Mnih, V et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[82] Mozer, MC. Neural net architectures for temporal sequence processing. In *Santa Fe Institute Studies in the Sciences of Complexity*, volume 15, pages 243–243. Addison-Wesley, 1993.

[83] Oord, A, Kalchbrenner, N, and Kavukcuoglu, K. Pixel recurrent neural networks. *arXiv e-prints*, 2016. arXiv:1601.06759.

[84] Pearlmutter, BA. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1(2):263–269, 1989.

[85] Pezeshki, M, Fan, L, Brakel, P, Courville, A, and Bengio, Y. Deconstructing the ladder network architecture. *arXiv e-prints*, 2015. arXiv:1511.06430.

[86] Polykovskiy, D et al. Entangled conditional adversarial autoencoder for de novo drug discovery. *Mol. Pharma.*, 15(10):4398–4405, 2018.

[87] Pomerleau, DA. ALVINN: An autonomous land vehicle in a neural network. In *NIPS*, pages 305–313, 1989.

[88] Press, WH, Flannery, BP, Teukolsky, SA, and Vetterling, WT. *Numerical Recipes in C*. Cambridge University Press, 1989.

[89] Radford, A, Metz, L, and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv e-prints*, 2015. arXiv:1511.06434.

[90] Randløv, J and Alstrøm, P. Learning to drive a bicycle using reinforcement learning and shaping. In *Proc. 15th Int. Conf. Machine Learning (ICML'98)*, pages 463–471, 1998.

[91] Rasmus, A, Berglund, M, Honkala, M, Valpola, H, and Raiko, T. Semi-supervised learning with ladder networks. In *NIPS*, pages 3546–3554, 2015.

[92] Rasmus, A., Valpola, H., Honkala, M., Berglund, M., and Raiko, T. Semi-supervised learning with ladder networks. *arXiv e-prints*, 2015. arXiv:1507.02672.

[93] Reddy, G, Celani, A, Sejnowski, TJ, and Vergassola, M. Learning to soar in turbulent environments. *Proc. Nat. Acad. Sci. USA*, 113(33):E4877–E4884, 2016.

[94] Rezende, DJ, Mohamed, S, and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. *arXiv e-prints*, 2014. arXiv:1401.4082.

[95] Rezende, DJ and Viola, F. Taming variational autoencoders. *arXiv e-prints*, 2018. arXiv:1810.00597.

[96] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[97] Rumelhart, DE, Hinton, GE, and Williams, RJ. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[98] Sabour, S, Frosst, N, and Hinton, GE. Dynamic routing between capsules. In *NIPS*, pages 3859–3869, 2017.

[99] Salakhutdinov, R and Larochelle, H. Efficient learning of deep Boltzmann machines. In *Proc. 13th Int. Conf. AI and Statistics*, pages 693–700, 2010.

[100] Samuel, AL. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

[101] Schäfer, A M and Zimmermann, HG. Recurrent neural networks are universal approximators. In *Int. Conf. Artificial Neural Networks*, pages 632–640. Springer, 2006.

[102] Schulman, J. *Optimizing Expectations: From Deep Reinforcement Learning to Stochastic Computation Graphs*. PhD thesis, University of California, Berkeley, 2016.

[103] Sejnowski, TJ. *The Deep Learning Revolution*. MIT Press, 2018.

[104] Sejnowski, TJ and Rosenberg, CR. NETtalk. *Complex Systems*, 1(1), 1987.

[105] Selfridge, OG. Pandemonium: A paradigm for learning. In *The Mechanisation of Thought Processes*. National Physical Laboratory, 1958.

[106] Shannon, CE. Programming a computer for playing chess. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 41(314):256–275, 1950.

[107] Shannon, CE and Weaver, W. *The Mathematical Theory of Communication*. University of Illinois Press, 1949.

[108] Silver, D et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–503, 2016.

[109] Silver, D et al. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[110] Simard, P, LeCun, Y, and Denker, J. Efficient pattern recognition using a new transformation distance. In *NIPS*, pages 50–58, 1992.

[111] Simeone, O. A brief introduction to machine learning for engineers. *arXiv e-prints*, 2017. arXiv:1709.02840.

[112] Sivia, DS. *Data Analysis: A Bayesian Tutorial*. Oxford University Press, 1996.

[113] Smolensky, P. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Department of Computer Science, University of Colorado at Boulder, 1986.

[114] Solla, SA, Sorkin, GB, and White, SR. Configuration space analysis for optimization problems. In Bienstock, E, editor, *Disordered Systems and Biological Organisation*, pages 283–293. Springer, 1986.

[115] Stone, JV. Connectionist models: Theoretical status, form, and function. *AISB*, 66, 1988.

[116] Stone, JV. Learning sequences with a connectionist model. Invited paper presented to the *Rank Prize Funds Symposium on Neural Networks*, February 1989.

[117] Stone, JV. The optimal elastic net: Finding solutions to the travelling salesman problem. *Proc. Int. Conf. Artificial Neural Networks*, pages 170–174, 1992.

[118] Stone, JV. Learning perceptually salient visual parameters through spatiotemporal smoothness constraints. *Neural Computation*, 8(7):1463–1492, 1996.

[119] Stone, JV. *Independent Component Analysis: A Tutorial Introduction*. MIT Press, 2004.

[120] Stone, JV. Distributed representations accelerate evolution of adaptive behaviours. *PLoS Comp. Biol.*, 3(8):e147, 2007.

[121] Stone, JV. *Vision and Brain*. MIT Press, 2012.

[122] Stone, JV. *Bayes' Rule: A Tutorial Introduction to Bayesian Analysis*. Sebtel Press, 2013.

[123] Stone, JV. *Information Theory: A Tutorial Introduction*. Sebtel Press, 2015.

[124] Stone, JV and Harper, N. Temporal constraints on visual learning. *Perception*, 28:1089–1104, 1999.

[125] Stone, JV, Hunkin, NM, and Hornby, A. Predicting spontaneous recovery of memory. *Nature*, 414:167–168, 2001.

[126] Sun, C, Shrivastava, A, Singh, S, and Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. In *IEEE Int. Conf. Computer Vision (ICCV)*, pages 843–852. IEEE, 2017.

[127] Sutskever, I, Vinyals, O, and Le, QV. Sequence to sequence learning with neural networks. *arXiv e-prints*, 2014. arXiv:1409.3215.

[128] Sutton, RS and Barto, AG. *Reinforcement Learning: An Introduction*, volume 1. MIT Press, 2018.

[129] Szegedy, C et al. Going deeper with convolutions. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 1–9. 2015.

[130] Tesauro, G. Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[131] Tishby, N, Pereira, FC, and Bialek, W. The information bottleneck method. *arXiv e-prints*, 2000. arXiv:physics/0004057.

[132] Tramel, EW, Gabrié, M, Manoel, A, Caltagirone, F, and Krzakala, F. Deterministic and generalized framework for unsupervised learning with restricted Boltzmann machines. *Phys. Rev. X*, 8:041006, 2018.

[133] Vincent, P, Larochelle, H, Lajoie, I, Bengio, Y, and Manzagol, P. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[134] Wang, H and Raj, B. On the origin of deep learning. *arXiv e-prints*, 2017. arXiv:1702.07800.

[135] Watkins, CJCH. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, 1989.

[136] Widrow, B and Hoff, ME. Adaptive switching circuits. *1960 WESCON Convention Record, Part IV*, pages 96–104, 1960.

[137] Williams, PM. Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, 7(1):117–143, 1995.

[138] Williams, RJ. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[139] Williams, RJ and Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[140] Wu, H and Gu, X. Towards dropout training for convolutional neural networks. *Neural Networks*, 71:1–10, 2015.

[141] Yuret, D. Knet: Beginning deep learning with 100 lines of Julia. In *Machine Learning Systems Workshop at NIPS*, 2016.

[142] Zemel, RS, Mozer, C, and Hinton, GE. TRAFFIC: Recognizing objects using hierarchical reference frame transformations. In Tourzesky, DS, editor, *Advances in Neural Information Processing Systems*, pages 266–273. Morgan Kaufmann, 1990.

Index

About the Author. Dr James Stone is an Honorary Reader in Vision and Computational Neuroscience at the University of Sheffield, UK.