

## Review



**Cite this article:** Smith MJ, Geach JE. 2023  
Astronomia ex machina: a history, primer and  
outlook on neural networks in astronomy. *R. Soc.  
Open Sci.* **10**: 221454.  
<https://doi.org/10.1098/rsos.221454>

Received: 9 November 2022

Accepted: 28 April 2023

### Subject Category:

Astronomy

### Subject Areas:

astrophysics/artificial intelligence

### Keywords:

neural networks, astrophysics, machine learning

### Authors for correspondence:

Michael J. Smith

e-mail: [mike@mjjsmith.com](mailto:mike@mjjsmith.com)

James E. Geach

e-mail: [j.geach@herts.ac.uk](mailto:j.geach@herts.ac.uk)

# Astronomia ex machina: a history, primer and outlook on neural networks in astronomy

Michael J. Smith and James E. Geach

Department of Physics, Astronomy and Mathematics, School of Physics, Engineering and  
Computer Science, University of Hertfordshire, Hatfield AL10 9AB, UK

MJS, 0000-0003-0220-5125; JEG, 0000-0003-4964-4635

In this review, we explore the historical development and future prospects of artificial intelligence (AI) and deep learning in astronomy. We trace the evolution of connectionism in astronomy through its three waves, from the early use of multilayer perceptrons, to the rise of convolutional and recurrent neural networks, and finally to the current era of unsupervised and generative deep learning methods. With the exponential growth of astronomical data, deep learning techniques offer an unprecedented opportunity to uncover valuable insights and tackle previously intractable problems. As we enter the anticipated fourth wave of astronomical connectionism, we argue for the adoption of GPT-like foundation models fine-tuned for astronomical applications. Such models could harness the wealth of high-quality, multimodal astronomical data to serve state-of-the-art downstream tasks. To keep pace with advancements driven by Big Tech, we propose a collaborative, open-source approach within the astronomy community to develop and maintain these foundation models, fostering a symbiotic relationship between AI and astronomy that capitalizes on the unique strengths of both fields.

## 1. Introduction

The concept of artificial intelligence (AI) can be traced back at least 350 years to Leibniz's *Dissertation on the Art of Combinations* [1]. Inspired by Descartes and Lull, Leibniz posited that, through the development of a 'universal language', all ideas could be represented by the combination of a small set of fundamental concepts, and that *new* concepts could be generated in a logical fashion, potentially by some computing machine. Leibniz's ambitious vision ('let us calculate') has not yet been realized, but the quest to emulate human reasoning, or at least to build a machine to mimic the computational and data processing capabilities of the human brain, has persisted to this day.

It might be fair to say that the roots of AI stretch even as far back as Lull's medieval philosophy that inspired Leibniz [2,3].

However, if we now consider AI to be a bona fide scientific discipline, then that discipline clearly emerged in the post-war years of the twentieth century, following Turing's simple enquiry 'can machines think?' [4]. Somewhat philosophical in nature, Turing's 1950 question succinctly articulates the ambition of AI, but from a nuts and bolts standpoint it took a further 5 years from Turing's query for what one might call the first AI program—the so-called 'Logic Theorist'—to be developed by Allen Newell, Cliff Shaw and Herbert Simon. Funded by the Research and Development (RAND) Corporation, the Logic Theorist was designed, in part, to emulate the role of a human mathematician, in that it could automate the proof of mathematical theorems. This was a breakthrough in computer science and the Logic Theorist was presented at the seminal Dartmouth Summer Research Project on Artificial Intelligence (DSRP AI) conference in 1956, now regarded as the true birth of AI as a field. Indeed, it was DSRPAI organizer John McCarthy who is credited with coining the term 'artificial intelligence' [5].

Natural intrigue—and clearly a good deal of fear—of the idea of AI has inspired popular culture no end, from Dick's *Do Androids Dream Of Electric Sheep?* to Crichton's *Westworld*, *Terminator's 'Skynet'* and beyond. Iain M. Banks's Galactic civilization known as 'The Culture' imagines a society run by powerful 'Minds' whose intelligence and wisdom far exceeds that of humans, and where biological beings and machines of equivalent sentience generally coexist peacefully, cooperatively and equitably. Science fiction notwithstanding, if these dreams are even possible, we are still years away from a machine that can genuinely think for itself [6,7]. Nevertheless, the question of how one mathematically (and algorithmically) models the workings and inter-relationships of biological neurons—neural networks—and the subsequent exploration of how they can find utility as tools in the data analyst's workshop is really what is being referred to when most people use the term 'AI' today.<sup>1</sup> While we must always be wary of hype and buzzwordism, it is the *application* of neural networks—and the possibility of tackling hitherto intractable problems—that offers genuine reason for excitement across many disparate fields of enquiry, including astronomy.

Astronomers have made use of artificial neural networks (ANNs) for over three decades. In 1994, Ofer Lahav, an early trailblazer, wryly identified the 'neuro-skeptics'—those resistant to the use of such techniques in serious astrophysics research—and argued that ANNs 'should be viewed as a general statistical framework, rather than as an estoteric approach' [8]. Unfortunately, this scepticism has persisted. This is despite the recent upsurge in the use of neural networks (and machine learning in general) in the field, as illustrated in figure 1. This scepticism also stands contrary to achievements within astronomy that would not be possible without the use of ANNs, such as photometric redshift estimation (e.g. [9,10]), astronomical object identification and clustering at scale (e.g. [11]) and entirely data-driven simulation (e.g. [12,13]). Most of the criticism of machine learning techniques, and deep learning<sup>2</sup> in particular, is levelled at the perceived 'black box' nature of the methodology. In this review, we provide a primer on how deep neural networks are constructed, and the mathematical rules governing their learning, which we hope will serve as a useful resource for neuro-skeptics. Nevertheless, we must recognize that a unified theoretical picture of how deep neural networks work does not yet exist. This remains a point of debate even within the deep learning community. For example, Yann LeCun responding to Ali Rahimi's 'Test of Time' award talk at the 31st Conference on Neural Information Processing Systems (NIPS) remarked:

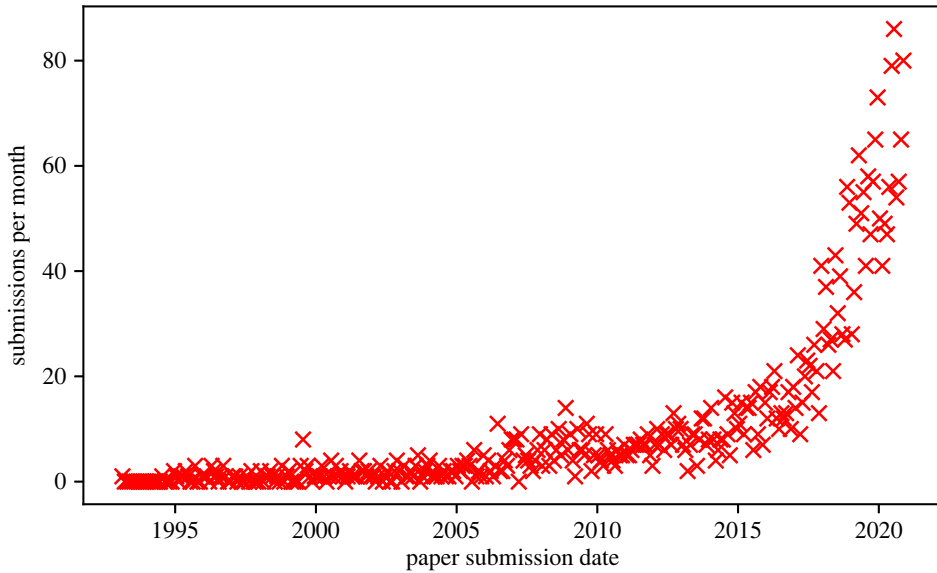
Ali gave an entertaining and well-delivered talk. But I fundamentally disagree with the message. The main message was, in essence, that the current practice in machine learning is akin to 'alchemy' (his word). It's insulting, yes. But never mind that: It's wrong! Ali complained about the lack of (theoretical) understanding of many methods that are currently used in ML, particularly in deep learning ... Sticking to a set of methods just because you can do theory about it, while ignoring a set of methods that empirically work better just because you don't (yet) understand them theoretically is akin to looking for your lost car keys under the street light knowing you lost them someplace else. Yes, we need better understanding of our methods. But the correct attitude is to attempt to fix the situation, not to insult a whole community for not having succeeded in fixing it yet. This is like criticizing James Watt for not being Carnot or Helmholtz [14].

Philosophical concerns aside, LeCun's fundamental point is that deep learning 'works' and therefore we should use it, even if we do not fully understand it. If one were being uncharitable, we could make similar arguments about the  $\Lambda$ CDM paradigm.

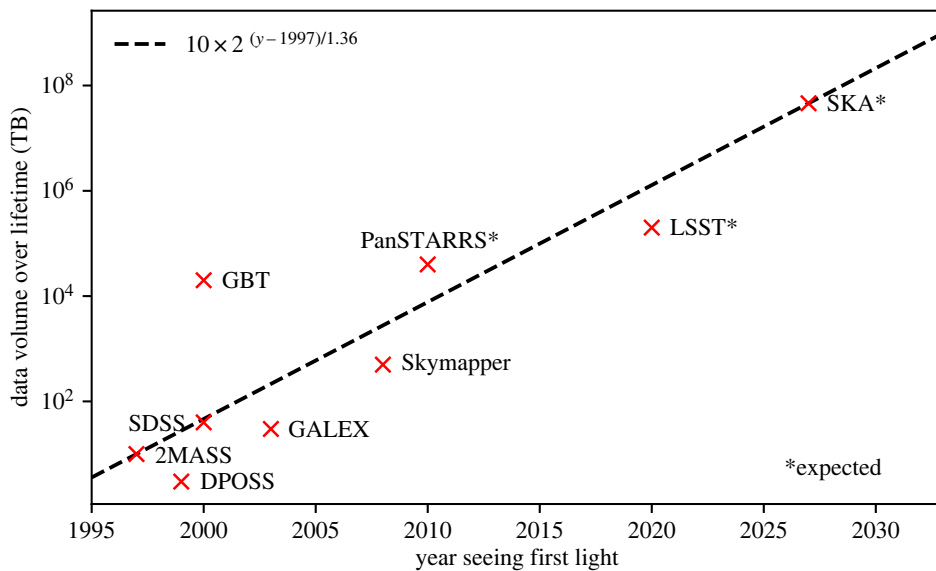
It is clear that in every field that deep learning has infiltrated we have seen a reduction in the use of specialist knowledge, to be replaced with knowledge automatically derived from data. We have already seen this process play out in many 'applied deep learning' fields such as computer Go [15], protein folding [16], natural language processing [17] and computer vision [18]. We argue that astronomy's data abundance corrals it onto a path no

<sup>1</sup>And the term is regularly misused, not only erroneously, but often cynically.

<sup>2</sup>Deep learning referring to the use of a network constructed of many layers of artificial neurons.



**Figure 1.** Here we see the number of arXiv:astro-ph submissions per month that have abstracts or titles containing one or more of the strings: ‘machine learning’, ‘ML’, ‘artificial intelligence’, ‘AI’, ‘deep learning’ or ‘neural network’. The raw data are in the public domain and are available at <https://www.kaggle.com/Cornell-University/arxiv>.



**Figure 2.** The data volume output of a selection of astronomical surveys over their lifetimes. We can see the astronomical survey data volume doubles every 16 months. Data are taken from Zhang & Zhao [19].

different to that trodden by other applied deep learning fields. This abundance is not a passing phase; the total astronomical data volume is already large and will increase exponentially in the coming years. We illustrate this in figure 2, where we present a selection of astronomical surveys and their estimated data volume output over their lifetimes [19]. And this is not even considering data associated with ever larger and more detailed numerical simulations (e.g. [20–22]). The current scale of the data volume already poses an issue for astronomy as many classical methods rely on human supervision and specialist expertise, and the increasing data volume will make exploring and exploiting these surveys through traditional human supervised and semi-supervised means an intractable problem. Of serious concern is the possibility that we will miss—or substantially delay—interesting and important discoveries simply due to our inability to accurately and consistently interrogate astronomical data at scale. Deep learning has shown great promise in automating information extraction in various data-intensive fields, and so is ideally poised as a solution to the challenge of processing ultra-large-scale astronomical data. But we do not need to stop there. This

review's outlook ventures a step further, and argues that astronomy's wealth of data should be considered a unique opportunity, and not merely an albatross.

Since astronomical connectionism's<sup>3</sup> humble beginnings in the late 1980s, there have been numerous excellent reviews on the application of artificial neural networks to astronomy (e.g. [23–25]). We take an alternative approach to previous literature reviews and survey the field holistically, in an attempt to paint astronomical connectionism's 'Big Picture' with broad strokes. While we cannot possibly include all works within astronomical connectionism,<sup>4</sup> we hope that this review serves as a historical background on astronomy's 'three waves' of increasingly automated connectionism, as well as presenting a general primer on neural networks that may assist those seeking to explore this fascinating topic for the first time.

In §§2 and 3, we explore initial work on multi-layer perceptrons within astronomy, where models required manually selected emergent properties as input. In §§4 and 5, we explore the second wave, which coincided with the dissemination of convolutional neural networks and recurrent neural networks—models where the multi-layer perceptron's manually selected inputs are replaced with raw data ingestion. In the third wave that is happening now we are seeing the removal of human supervision altogether with deep learning methods inferring labels and knowledge directly from the data, and we explore this wave in §§6–8. Finally, in §9, we look to the future and predict that we will soon enter a fourth wave of astronomical connectionism. We argue that if astronomy follows the pattern of other applied deep learning fields we will see the removal of expertly crafted deep learning models, to be replaced with fine-tuned versions of an all-encompassing 'foundation' model. As part of this fourth wave, we argue for a symbiosis between astronomy and connectionism, a symbiosis predicated on astronomy's relative data wealth and deep learning's insatiable data appetite. Many ultra-large datasets in machine learning are proprietary or of poor quality, and so there is an opportunity for astronomers as a community to develop and provide a high-quality multi-modal public dataset. In turn, this dataset could be used to train an astronomical foundation model to serve state-of-the-art downstream tasks. Owing to foundation models' hunger for data and compute, a single astronomical research group could not bring about such a model alone. Therefore, we conclude that astronomy as a discipline has slim chance of keeping up with a research pace set by the Big Tech goliaths—that is, unless we follow the examples of EleutherAI and HuggingFace and pool our resources in a grassroots open-source fashion.

Before moving on, we must first admit to our readers that we have not been entirely honest with them. The abstract of this review has not been written by us. It was generated by prompting OpenAI's generative pretrained transformer 4 ('GPT-4') neural network-based foundation model with this paper's introduction [26,27]. To be precise, we prompted the GPT-4 engine provided by 'ChatGPT Plus' with all the text in §1 up until this paragraph in raw LaTeX format. We then appended the following prompt to the introduction text:

Write an abstract for the above text that will catch the reader's eye, and make them interested in the paper. Make the abstract 160 words or less, and touch on the value of GPT-like models in astronomy.

We did not alter the GPT-generated output whatsoever. We explore these foundation models and their possible astronomical uses in more detail in §9.

## 2. A primer on artificial neurons

In 1943 McCulloch & Pitts [28] proposed the first computational model of a biological neuron (MP neuron; [28]). Their model consisted of a set of binary inputs  $x_i \in \{0, 1\}$  and a single binary output  $y \in \{0, 1\}$ . Their model also defines a single 'inhibitory' input  $\mathcal{I} \in \{0, 1\}$  that blocks output if  $\mathcal{I} = 1$ . If the sum of the inputs exceeds a threshold value  $\Theta$ , the MP neuron 'fires' and outputs  $y = 1$ . Mathematically, we can write the MP neuron function as

$$\text{MP}(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{i=1}^n x_i > \Theta \text{ and } \mathcal{I} = 0, \\ 0 & \text{otherwise.} \end{cases}$$

The MP neuron is quite a powerful abstraction. Single MP neurons can calculate simple Boolean functions, and more complicated functions can be calculated when many MP neurons are chained together. However, there is one show-stopping issue: the MP neuron is missing the capacity to learn.

<sup>3</sup>Since its inception, AI research can be broadly categorized into two schools: 'symbolic' and 'connectionist'. Symbolists see the mind as a collection of fully formed representations, and attempt to mimic human reasoning through a logical rule-based processing of these symbols. This approach contrasts with connectionist (or neural network-based) AI, which takes a bottom-up approach and simulates cognition by mimicking the way neurons in the human brain work.

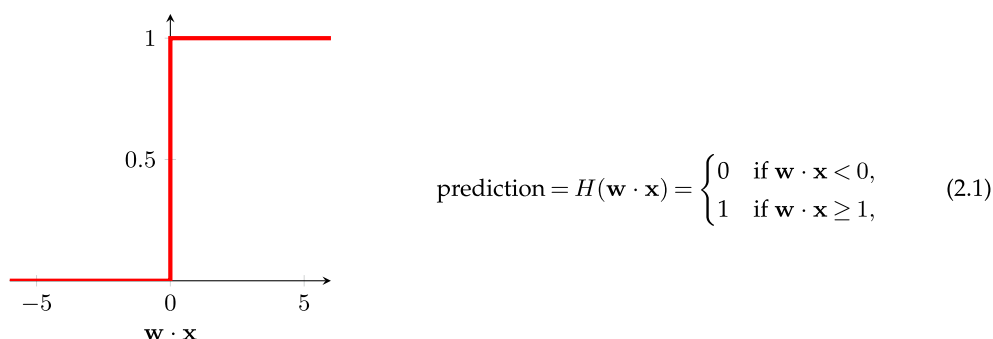
<sup>4</sup>We refer the reader to figure 1!

Rosenblatt [29] addressed this by combining the MP neuron with Hebb's neuronal wiring theory<sup>5</sup> [30], and we will explore a related training formulation in the next subsection.

## 2.1. The perceptron

This subsection aims to provide the reader a foundation and intuition for the gradient-based learning that dominates contemporary neural network architectures. Therefore, we diverge from Rosenblatt's original learning algorithm and instead describe a gradient-based training algorithm. The interested reader will find an analysis of Rosenblatt's original learning algorithms in the 'Mathematical analysis of learning in the perceptron' section of Rosenblatt [29].

Like the MP neuron, the perceptron takes a number of numeric inputs ( $x_i$ ). However, unlike the MP neuron, each one of these inputs is multiplied by a corresponding weight ( $w_i$ ) signifying the importance the perceptron assigns to a given input. As shown in figure 3, we can then sum this list of products and pass it into an 'activation function'. Let us use the Heaviside step function as our activation function,



where  $\mathbf{x}$  is a set of inputs, and  $\mathbf{w}$  is a set of 'weights' that represent the importance of each input.

To concretize how we could train our perceptron, we will use an example. Let us say that we want to automatically label a set of galaxy images as either 'spiral' or 'elliptical'. To do this, we first need to compile a training dataset of galaxy images. This training set would consist of spiral and elliptical galaxies, and each image would have a ground truth label  $y$ —say '0' for a spiral galaxy and '1' for an elliptical. To train our perceptron, we randomly choose one image from the training set, and feed it to the perceptron, with the numerical value of each pixel corresponding to an input  $\{x_1, \dots, x_N\}$ . These inputs are multiplied by their corresponding weight  $\{w_1, \dots, w_N\}$ . A bias term ( $b = w_0 x_0$ , where  $x_0 = 1$ ) is also added to the inputs, which allows the neuron to shift its activation function linearly. Since we do not want our perceptron to have any prior knowledge of the task, we initialize the weights at random. The resulting products are then summed. Finally, our activation function  $H$  transforms  $\mathbf{w} \cdot \mathbf{x}$  and produces a prediction  $p$ . We then compare  $p$  with  $y$  via a 'loss function,' which is a function that measures the difference between  $p$  and  $y$ . The loss can be any differentiable function, so for illustration purposes we will define it here as the L1 loss:  $\mathcal{L}(y, p) = |y - p|$ . Now that we can compare with the ground truth, we need to work out how a change in one of our weights affects the loss (that is, we want to find  $\partial \mathcal{L} / \partial \mathbf{w}$ ). We can calculate this change with the chain rule

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}}{\partial p} \frac{\partial p}{\partial \mathbf{w}}, \quad (2.2)$$

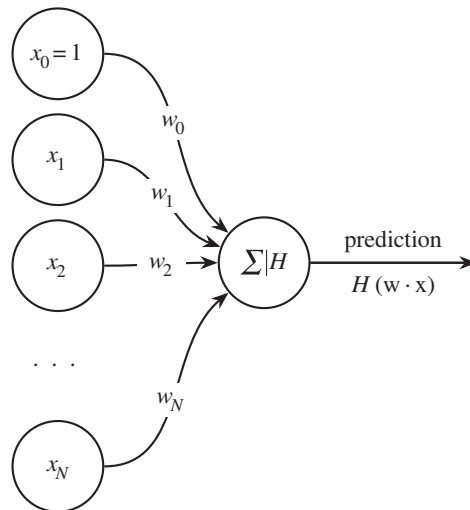
and since  $p = H(\mathbf{w} \cdot \mathbf{x})$  and  $\partial p / \partial \mathbf{w} = H' \mathbf{x}^T$  we get

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}}{\partial p} \odot (H' \mathbf{x}^T),$$

where  $\odot$  is the distributive Hadamard product. Thus, we can update the weights to decrease the loss function,

$$\begin{aligned} \mathbf{w}_{\text{next}} &= \mathbf{w} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{w}} \\ &= \mathbf{w} - \eta \frac{\partial \mathcal{L}}{\partial p} \odot (H' \mathbf{x}^T), \end{aligned}$$

<sup>5</sup>Also known by the mantra 'cells that fire together wire together'.



**Figure 3.** A single neuron (or perceptron) with a bias  $w_0$ , inputs  $x_1, x_2, \dots, x_N$ , and weights  $w_1, w_2, \dots, w_N$ .

where  $\eta$  is the learning rate.<sup>6</sup> If we repeat this process our perceptron will get better and better at classifying our galaxies!

While we provide the above example for illustrative purposes, we will need a more powerful algorithm to produce a useful classifier of galaxy morphology. This need is perhaps most famously discussed in *Perceptrons: An Introduction to Computational Geometry* ([31], e.g. §13.0). Minsky & Papert show that the single-layer perceptron is only able to calculate linearly separable functions, among other limitations. Their book (alongside a consensus that AI had failed to deliver on its early grandiose promises) delivered a big blow to the connectionist school of artificial intelligence.<sup>7</sup> In the years following Minsky & Papert [31], governmental and industry funding was pulled from connectionist research laboratories, ushering in the first ‘AI winter’.<sup>8</sup>

Yet, as exemplified in Rosenblatt ([36], §5.2, theorem 1) it was known at the time that multi-layer perceptrons could calculate nonlinearly separable functions (such as the ‘exclusive or’). We can prove intuitively that a set of neurons can calculate *any* function: a perceptron can perfectly emulate a NAND gate (figure 4), and the singleton set {NAND} is functionally complete. Since we can combine a set of NAND gates to calculate any function, *we must also be able to combine a set of neurons to calculate any function*. This result is also explored in a more formal proof by both Cybenko [37] and Hornik *et al.* [38]. They show that an infinitely wide neural network can calculate any function. Similarly, Lu *et al.* [39] show that an infinitely deep neural network is a universal approximator. Such a group of neurons is known as the multi-layer perceptron (MLP). Unfortunately, we cannot simply stack perceptrons together as we are missing one vital ingredient: a way to train the network! At the time of Minsky & Papert’s treatise on perceptrons, there was no widely known algorithm (in the West; see [34]) that could train such a multi-layer network. In Minsky & Papert’s own words:

Nevertheless, we consider it to be an important research problem to elucidate (or reject) our intuitive judgment that the extension [from one layer to many] is sterile. Perhaps some powerful convergence theorem will be discovered, or some profound reason for the failure to produce an interesting ‘learning theorem’ for the multilayered machine will be found. (Minsky & Papert [31], §13.2 on MLPs)

The field had to wait almost two decades for such an algorithm to become widespread. In the next subsection, we will explore backpropagation, the algorithm that ultimately proved Minsky and Papert’s intuition wrong.

## 2.2. The multi-layer perceptron

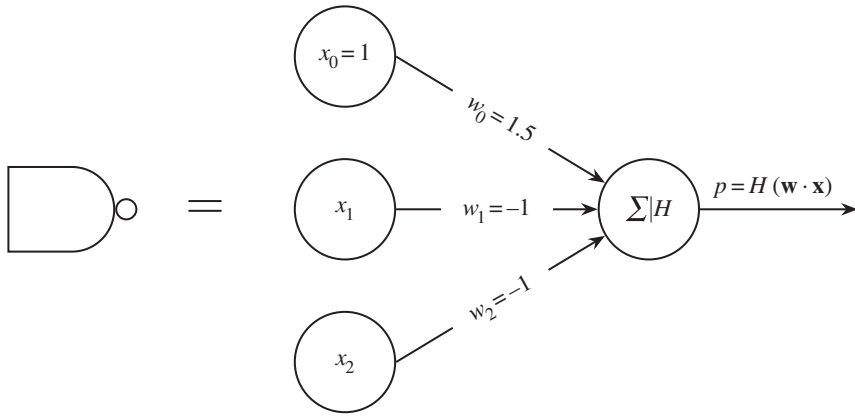
Grouping many artificial neurons together may result in something resembling figure 5. This network consists of an input layer, two intermediate ‘hidden’ layers, and an output layer. As in the previous

<sup>6</sup>The eagle-eyed reader may have noticed that since the derivative of the Heaviside step function is the Dirac delta function, we will only update the perceptron’s weights on an incorrect prediction. If we want to also learn from positive examples, we need to use a smoothly differentiable activation function. This is explored in the next subsection.

<sup>7</sup>See Olazaran [32] and Metz [33] for a closer look at the conflicts and personalities that shaped AI.

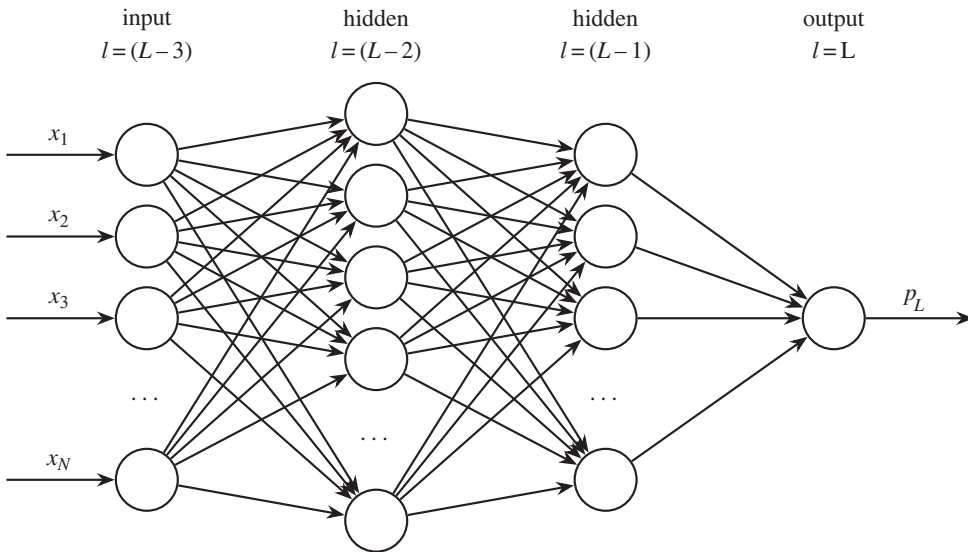
<sup>8</sup>At least, in the Western world. Connectionism continued in earnest in the Soviet Union [34,35].





$x_1$	$x_2$	$\neg(x_1 \wedge x_2)$	$p = H(\mathbf{w} \cdot \mathbf{x})$
0	0	1	$H(1.5 + (-1) \cdot 0 + (-1) \cdot 0) = 1$
0	1	1	$H(1.5 + (-1) \cdot 0 + (-1) \cdot 1) = 1$
1	0	1	$H(1.5 + (-1) \cdot 1 + (-1) \cdot 0) = 1$
1	1	0	$H(1.5 + (-1) \cdot 1 + (-1) \cdot 1) = 0$

**Figure 4.** If we define  $H(\mathbf{w} \cdot \mathbf{x})$  as in equation (2.1), we can set a perceptron’s weights so that it is equivalent to the NAND gate.

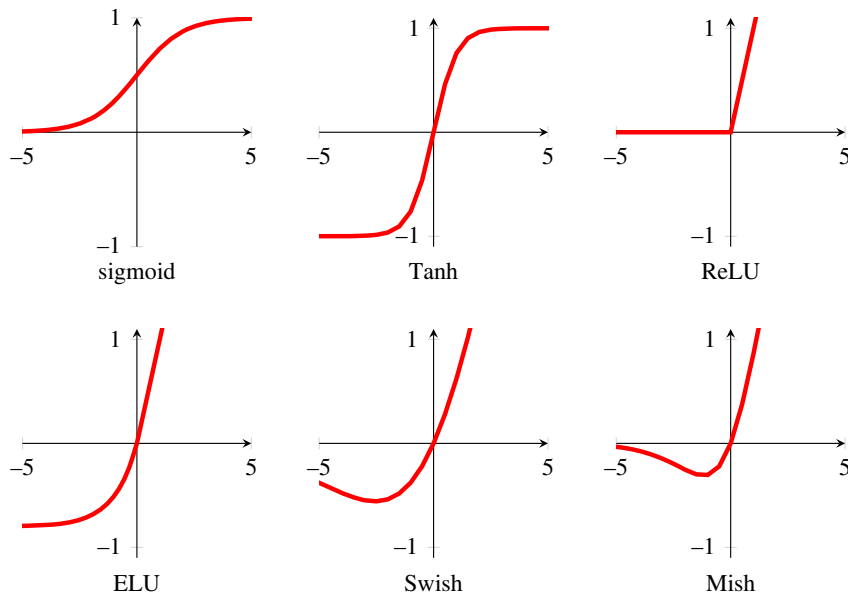


**Figure 5.** The multi-layer perceptron, or artificial neural network. The depicted network has two hidden layers. It takes  $N$  inputs  $x_1, x_2, \dots, x_N$ , and outputs a prediction  $p_L$ . Note that here we omit the explicit bias terms (i.e.  $w_0$ ).

section, let us say that we want a classifier that can classify a set of galaxy images into elliptical and spiral types. In an MLP similar to figure 5, a neuron would be assigned to each pixel in a galaxy image. Each neuron would take the numeric value of that pixel, and propagate that signal forward into the network. The next layer of neurons does the same, with the input being the previous layer’s output. This process continues until we reach the output layer. In a binary classification task like our galaxy classifier, this layer outputs a value between zero and one. Thus, if we define a spiral galaxy as zero, and an elliptical galaxy as one, we would want the network output to be near zero for a spiral galaxy input (and vice versa).

In §2.1, we found the change we needed to apply to a single neuron’s weights to make it learn from a training example. We can train an MLP in a similar way by employing the reverse mode of automatic differentiation (or backpropagation) to learn from our galaxy training dataset [40–42].<sup>9</sup> We want our

<sup>9</sup>Some controversy surrounds backpropagation’s discovery. The Finnish computer scientist Linnainmaa proposed the reverse mode of automatic differentiation and adapted the algorithm to run on computers in their 1970 (Finnish language) thesis [43]. They first



**Figure 6.** A curated selection of activation functions. In all plots, the  $x$ -axis is the input, and the  $y$ -axis is the output. The rectified linear unit (ReLU) activation function was first introduced in the context of neural networks in Fukushima [46] and later rediscovered, named and popularized in Nair & Hinton [47]. The exponential linear unit (ELU), Swish and Mish activations were, respectively, introduced in Clevert *et al.* [48], Ramachandran *et al.* [49] and Misra [50].

network to learn when it makes both a correct and incorrect prediction, so we define our activation function as a smoothed version of the Heaviside step function. This ensures that a signal is present in the derivative no matter which values are input. This activation function is known as the ‘sigmoid’ function, and is shown in figure 6. As in §2.1, we define a loss function  $\mathcal{L}(y, p)$  that describes the similarity between a ground truth ( $y$ ) and a prediction ( $p$ ). We also define a neuron’s activation function as  $\varphi(\mathbf{w} \cdot \mathbf{x})$  where  $\mathbf{w} \cdot \mathbf{x}$  is the weighted sum of a neuron’s inputs. Following from equation (2.2)

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_l} = \frac{\partial \mathcal{L}}{\partial p_l} \frac{\partial p_l}{\partial \mathbf{w}_l},$$

where  $l$  is a layer in the MLP. In the same way as in §2.1, we can calculate an MLP’s final layer’s ( $l = L$ ) weight updates in terms of known values

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_L} = \frac{\partial \mathcal{L}}{\partial p_L} \odot (\varphi'_L \mathbf{p}_{L-1}^T), \quad (2.3)$$

where  $\mathbf{p}_{L-1}$  are the outputs from the previous layer. To calculate the  $(L - 1)$ th layer’s weight updates, we use the chain rule

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{L-1}} = \frac{\partial \mathcal{L}}{\partial p_L} \frac{\partial p_L}{\partial \mathbf{p}_{L-1}} \frac{\partial \mathbf{p}_{L-1}}{\partial \mathbf{w}_{L-1}}.$$

Likewise for the  $(L - n)$ th layer

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{L-n}} = \frac{\partial \mathcal{L}}{\partial p_L} \left( \prod_{i=1}^n \frac{\partial \mathbf{p}_{L+1-i}}{\partial \mathbf{p}_{L-i}} \right) \frac{\partial \mathbf{p}_{L-n}}{\partial \mathbf{w}_{L-n}}.$$

Now we can start plugging in some known values. Since  $\mathbf{p}_l = \varphi(\mathbf{w}_l \cdot \mathbf{p}_{l-1})$ , it follows that  $\partial \mathbf{p}_l / \partial \mathbf{p}_{l-1} = \varphi'_l \mathbf{w}_l^T$ , and  $\partial \mathbf{p}_l / \partial \mathbf{w}_l = \varphi'_l \mathbf{p}_{l-1}^T$ . So

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{L-n}} = \frac{\partial \mathcal{L}}{\partial p_L} \odot \left( \prod_{i=1}^n \varphi'_{L-i} \mathbf{w}_{L-i}^T \right) (\varphi'_{L-n} \mathbf{p}_{L-n-1}^T). \quad (2.4)$$

published their findings in English in 1976. Werbos [41] then proposed applying an adaptation of Linnainmaa’s method to artificial neural networks. Rumelhart *et al.* [42] showed experimentally that backpropagation can generate meaningful internal representations within a neural network, and popularized the method. Here we will err on the side of caution and cite all three manuscripts. For further reading, we recommend Schmidhuber [44] and Baydin *et al.* [45].



Combining equation (2.3) with equation (2.4) we get the weight update algorithm for the  $(L - n)$ th layer of the MLP

$$\mathbf{w}_{\text{next}} = \mathbf{w} - \eta \begin{cases} \frac{\partial \mathcal{L}}{\partial p_L} \odot (\boldsymbol{\phi}'_L \mathbf{P}_{L-1}^T), & \text{for } n = 0, \\ \frac{\partial \mathcal{L}}{\partial p_L} \odot \left( \prod_{i=1}^n \boldsymbol{\phi}'_{L-i} \mathbf{w}_{L-i}^T \right) (\boldsymbol{\phi}'_{L-n} \mathbf{P}_{L-n-1}^T), & \text{for } n > 0. \end{cases} \quad (2.5)$$

With this equation<sup>10</sup> in hand, we can use the same technique described earlier in this section and in §2.1 to update the network's weights with each galaxy image to decrease the loss function  $\mathcal{L}$ . Again, as  $\mathcal{L}$  is minimized, our MLP will classify our elliptical and spiral galaxy images with increasing accuracy.

### 3. Astronomy's first wave of connectionism

Connectionism was first discussed within astronomy in the late 1980s, after the popularization of backpropagation (see footnote 9) and the consequent passing of the first 'AI winter'. Two radical studies emerged in 1988 that recognized areas where astronomy could benefit from the use of ANNs [51,52]. Together, they identified that astronomical object classification,<sup>11</sup> and telescope scheduling could be solved through the use of an ANN. These studies were followed by a rapid broadening of the field, and the application of connectionism to many disparate astronomical use cases ([23] and references therein). In this section, we will outline areas where MLPs found an early use in astronomy.

#### 3.1. Classification problems

Odehahn *et al.* [53] classified astronomical objects into star and galaxy types. These were taken from the Palomar Sky Survey Automated Plate Scanner catalogue [54]. To compile their dataset, they first extracted a set of emergent image parameters from the scanned observations. These parameters included the diameter, ellipticity, area and plate transmission. The parameters were then used to train both a linear perceptron and a feedforward MLP to classify the objects into stars or galaxies. Odehahn *et al.* [53] found that their best performing model could classify galaxies with a completeness of 95% for objects down to a magnitude less than 19.5. This work was followed by many more studies on the star/galaxy classification problem (e.g. [55–58]). Galaxy morphological type classification was explored in the early 1990s. Storrie-Lombardi & Lahav [59] describe an MLP that takes as input a selected set of 13 galaxy summary statistics, and uses this information to classify a galaxy into one of five morphological types. Storrie-Lombardi & Lahav [59] report a top one accuracy of 64%, and a top two accuracy of 90%. This pilot study was followed by several studies from the same group that confirmed that MLPs are effective automatic galaxy morphological classifiers ([60–65], see §5 for a continuation of this line of research).

MLPs were also used in other classification tasks; here we highlight a few further areas where MLPs were applied. Von Hippel *et al.* [66] classified stellar spectra into temperature types, and Klusch & Napiwotzki [67] did the same for Morgan–Keenan system types. Chon [68] described the use of an MLP to search for and classify muon events (and therefore neutrino observations) in the Sudbury Neutrino Observatory. Quasar classification has been explored in several studies [69–71]. Seminally, Carballo *et al.* [69] used an MLP to select quasar candidates given their radio flux, integrated-to-peak flux ratio, photometry and point spread function in the red and blue bands, and their radio-optical position separation. They found good agreement between their model and that of the decision tree described in White *et al.* [72], confirming MLPs as a competitive alternative to more traditional machine learning. As part of the Supernova Photometric Classification Challenge (SPCC, [73]), Karpenka *et al.* [74] proposed the use of a neural network to classify supernovae into Type-1a/non-Type-1a classes. To classify their light curves, they first used a hand-crafted fitting function, and then trained their MLP on the fitted coefficients. They found that their model was competitive with other, more complex models trained on the SPCC dataset. From the studies discussed in this section, we can safely conclude that MLPs are effective classifiers of astronomical data, when given important parameters extracted by an expert guide.

<sup>10</sup>If we examine equation (2.5) carefully, we can see why we add nonlinearities between the MLP layers; without activation functions equation (2.5) collapses to the equivalent of a single layer MLP!

<sup>11</sup>Specifically, galaxies were discussed in Rappaport & Anderson [51] and point sources observed with the Infra-Red Astronomical Satellite (IRAS) were discussed in Adorf & Johnston [52].

## 3.2. Regression problems

MLPs were also used in regression problems. Angel *et al.* [75] applied them first to adaptive telescope optics. They trained their MLP on 250 000 simulated in focus and out of focus observations of stars as seen by the Multiple Mirror Telescope (MMT). From the flattened  $13 \times 13$  pixel observations, their network predicted the piston position and tilt required for each of the MMT's mirrors to bring the stars into focus. After the application of these corrections, the authors were able to recover the original profile. In follow-up studies, Sandler *et al.* [76] and Lloyd-Hart *et al.* [77] proved that Angel *et al.*'s MLP worked on the real MMT.

Photometric redshift estimation was explored in many concurrent studies (e.g. [9,10,65,78,79]). Firth *et al.* [10] trained a neural network to predict the redshift of galaxies contained in the Sloan Digital Sky Survey (SDSS) early data release [80]. The galaxies were input to the neural network as a set of summary parameters, and the output was a single float representing the galaxy redshift. They found their network attained a performance comparable to classical techniques. Extending and confirming the work by Firth *et al.* [10], Ball *et al.* [65] used an MLP to predict the redshift of galaxies contained in the SDSS's first data release [81]. They also showed that MLPs were capable of predicting the galaxies' spectral types and morphological classifications.

Of course, MLPs have been used more widely in astronomical regression tasks. Here we will cherry pick a few studies to show the MLP's early breadth of use. Sunspot maxima prediction was carried out by Koons & Gorney [82]. They found their MLP-based method was capable of predicting the number of sunspots when trained on previous cycles. Bailer-Jones *et al.* [83] predicted the effective temperature of a star from its spectrum. Auld *et al.* [84,85] applied MLPs to cosmology, demonstrating that MLPs are capable of predicting the cosmic microwave background power spectra and matter power spectra when given a set of cosmological parameters. Nørgaard-Nielsen & Jørgensen [86] used an MLP to remove the foreground from microwave temperature maps. From the studies discussed in this section, we can see that MLPs are effective regressors of astronomical data, when given significant parameters extracted by an expert guide.

## 4. Contemporary supervised deep learning

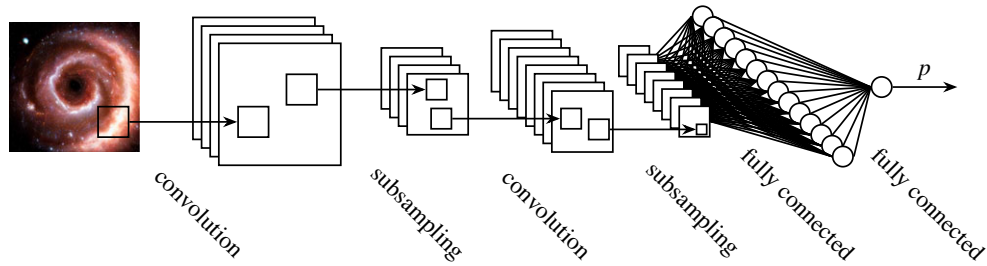
There are some issues with MLPs. Primarily they do not scale well to high-dimensional datasets. For example, if our dataset consists of images with  $128 \times 128$  pixels, we will need 16 384 neurons in the MLP's input layer alone! As we move into the hidden layers, this scaling issue only gets worse. Also, since MLPs must take an unrolled image as an input, they disregard any spatial properties of their training images, and so either need a substantial amount of training data to classify or generate large images,<sup>12</sup> or an expert to extract descriptive features from the data in a preprocessing step. We can see this issue writ large in the previous section—most of the MLP applications described in §3 require an expert to extract features from the data for the network to then train on! This drawback is not ideal; what if there are features within the raw data that are not present in these cherry-picked statistics? In that case, it would be preferable to let the neural network take in the raw data as input, and then learn which features are the most descriptive. We will discuss neural network architectures that solve both the MLP scaling problem and the expert reliance problem in this section. After we have explored these architectures in general, we will discuss their application to astronomical problems in §5.

### 4.1. Convolutional neural networks

Unlike the MLP described in the previous section, convolutional neural networks (CNNs; introduced in Fukushima [46] and first combined with backpropagation in LeCun *et al.* [93]) do not entirely consist of fully connected layers, where each neuron is connected to every neuron in the previous and subsequent layers. Instead, the CNN (such as the one depicted in figure 7) uses convolutional layers in place of the majority (or all) of the dense layers.

We can think of a convolutional layer as a set of learnt 'feature filters'. These feature filters perform a local transform on input imagery. In classical computer vision, these filters are hand crafted, and perform a predetermined function, such as edge detection or blurring. By contrast, a CNN learns the optimal set

<sup>12</sup>At the height of the convolutional neural network architecture's popularity in the mid-2010s, these were real problems. However, with the growth of computing power and data in recent years we are seeing a resurgence of the more general MLP model (e.g. [87–90]). This follows the prevailing trend in AI where the removal of human-crafted features and biases ultimately results in more expressive models that learn such features and biases directly from data [91,92].



**Figure 7.** A convolutional neural network classifying a spiral galaxy image.<sup>13</sup>

of filters for its task (say, galaxy classification). Equation (4.1) shows two different convolution<sup>14</sup> operators being performed on an array.

$$\begin{bmatrix} 39 & 57 & 86 & 9 & 26 \\ 90 & 74 & 63 & 87 & 98 \\ 79 & 34 & 26 & 16 & 46 \\ 67 & 61 & 96 & 1 & 79 \\ 33 & 47 & 15 & 49 & 29 \end{bmatrix} \star \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 26 & 16 & 46 \\ 96 & 1 & 79 \\ 15 & 49 & 29 \end{bmatrix} \quad (4.1)$$

$$\begin{bmatrix} 39 & 57 & 86 & 9 & 26 \\ 90 & 74 & 63 & 87 & 98 \\ 79 & 34 & 26 & 16 & 46 \\ 67 & 61 & 96 & 1 & 79 \\ 33 & 47 & 15 & 49 & 29 \end{bmatrix} \star \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 139 & 136 & 219 \\ 220 & 101 & 158 \\ 155 & 179 & 56 \end{bmatrix}$$

In the above equation, the operation is represented as a matrix. In a CNN, the matrix is a set of neuronal weights. As shown in figure 7, there are multiple feature maps in a convolutional layer, each containing a set of weights independent to the other feature maps, and learning to extract a different feature. Owing to the convolution operator's inbuilt translational equivariance, these features can be detected by the convolutional layer no matter where they are in the image. As in the MLP described in the previous section, the weights are updated using backpropagation to minimize a loss function. We will discuss astronomical applications of CNNs in §5, after we introduce modern CNN architectures.

## 4.2. Recurrent neural networks

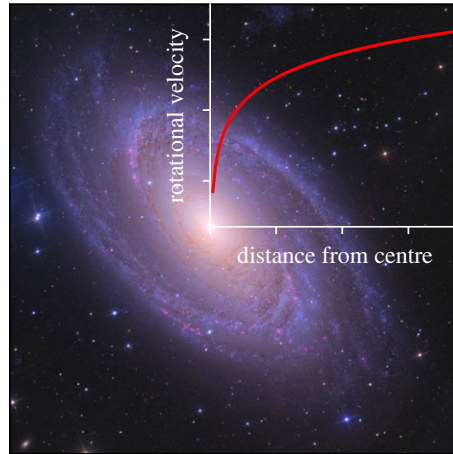
Standard feedforward neural networks like the MLP (§2.2) and CNN (§4.1) generate a fixed-size vector given a fixed-size input.<sup>15</sup> But, what if we want to classify or generate a variably sized vector? For example, we might want to classify a galaxy's morphology given its rotation curve. A rotation curve describes the velocity of a galaxy's visible stars versus their distance from the galaxy's centre. Figure 8 shows a possible rotation curve for Messier 81. A rotation curve's length depends on the size of its galaxy, and due to this variable length, and the fact that MLPs take a fixed-size input, we cannot easily use an MLP for classification. Recurrent neural networks (RNNs), however, can take a variable length input and produce a variable length output. An RNN differs from a feedforward MLP by having a hidden state that acts as a 'memory' store of previously seen information. As the RNN encounters new data, its weights are altered through the backpropagation through time algorithm (BPTT; [97] and references therein. Also see footnote 9).

We can use an RNN similar to figure 9 to classify our rotation curves. We express the rotation curve as a list  $\{x_1, x_2, \dots, x_N\}$ , with each  $x$  being a measurement of the rotational velocity at a certain radius. Then we feed this list into the RNN sequentially in the same way as shown in figure 9. The RNN will produce an output for each  $x$  fed to it, but we ignore those until we feed in  $x_N$ , the rotational velocity furthest from the galaxy's centre. When we feed in  $x_N$ , the RNN produces a prediction  $p_N$ , which we can then compare with a ground truth  $y_N$  via a loss function  $\mathcal{L}_N$ . In our case,  $y$  is an integer label representing the galaxy's morphological class. The comparison  $\mathcal{L}_N(y_N, p_N)$  is a function that represents the distance between the RNN prediction and the ground truth. We can then reduce  $\mathcal{L}_N(y_N, p_N)$  by updating the RNN's

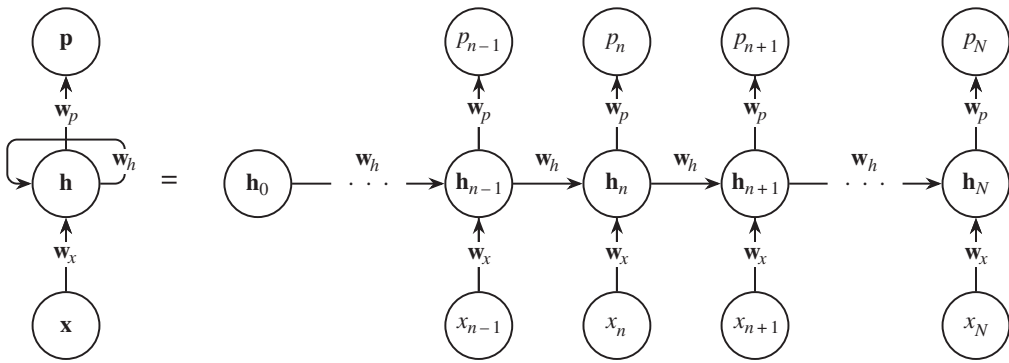
<sup>13</sup>All astronomical objects shown in the neural network diagrams within this manuscript are generated via text prompts fed into a latent diffusion neural network model [94].

<sup>14</sup>We must note that in equation (4.1) we follow most deep learning libraries and perform a cross-correlation and *not* a convolution. However, since the weights are learnt, this does not matter; the neural network will simply learn a flipped representation of the cross-correlation.

<sup>15</sup>As with any rule there are exceptions, such as CNNs containing a global average pooling layer [95].



**Figure 8.** An example of a galaxy rotation curve, plotted over an image of Messier 81 [96].



**Figure 9.** A recurrent neural network with weights  $\{\mathbf{w}_x, \mathbf{w}_p, \mathbf{w}_h\}$ , a hidden state  $\mathbf{h}_n$ , inputs  $\mathbf{x}$  and a prediction  $p_{n=N}$  is unrolled into its constituent processes.

weights through BPTT so that the weights  $\{\mathbf{w}_x, \mathbf{w}_p, \mathbf{w}_h\}$  follow  $\nabla \mathcal{L}_N$  downwards. As we do this, our RNN will improve its galaxy classifications.

BPTT's mathematical derivation is akin to the one we explored in §2.2, and we will quickly derive it here for posterity. Let us first look at the forward propagation equations,

$$\begin{aligned}\mathcal{L}_n &= |y_n - p_n|, \\ p_n &= \varphi(\mathbf{w}_p \cdot \mathbf{h}_n) \\ \text{and} \quad \mathbf{h}_n &= \phi(\mathbf{w}_h \cdot \mathbf{h}_{n-1} + \mathbf{w}_x \cdot \mathbf{x}_n).\end{aligned}$$

From these we see that we need to express  $\partial \mathcal{L}_n / \partial \mathbf{w}_p$ ,  $\partial \mathcal{L}_n / \partial \mathbf{w}_h$  and  $\partial \mathcal{L}_n / \partial \mathbf{w}_x$  as known values to train the network.  $\partial \mathcal{L}_n / \partial \mathbf{w}_p$  is relatively easy; via the chain rule, and the fact that  $\partial p_n / \partial \mathbf{w}_p = \varphi' \mathbf{h}_n^T$

$$\begin{aligned}\frac{\partial \mathcal{L}_n}{\partial \mathbf{w}_p} &= \frac{\partial \mathcal{L}_n}{\partial p_n} \frac{\partial p_n}{\partial \mathbf{w}_p}, \\ &= \frac{\partial \mathcal{L}_n}{\partial p_n} \odot \varphi' \mathbf{h}_n^T.\end{aligned}\quad (4.2)$$

$\partial \mathcal{L}_n / \partial \mathbf{w}_h$  is more tricky, so we will go step by step. We already know that

$$\frac{\partial \mathcal{L}_n}{\partial \mathbf{w}_h} = \frac{\partial \mathcal{L}_n}{\partial p_n} \frac{\partial p_n}{\partial \mathbf{h}_n} \frac{\partial \mathbf{h}_n}{\partial \mathbf{w}_h}.\quad (4.3)$$

However, we see in figure 9 that  $\mathbf{h}_n$  depends on  $\mathbf{h}_{n-1}$ , which depends on  $\mathbf{h}_{n-2}$  (and so on). We also notice that all the hidden states depend on  $\mathbf{w}_h$ . We therefore rewrite equation (4.3) to make this explicit,

$$\begin{aligned}\frac{\partial \mathcal{L}_n}{\partial \mathbf{w}_h} &= \frac{\partial \mathcal{L}_n}{\partial p_n} \frac{\partial p_n}{\partial \mathbf{h}_n} \sum_{j=1}^n \frac{\partial \mathbf{h}_n}{\partial \mathbf{h}_j} \frac{\partial \mathbf{h}_j}{\partial \mathbf{w}_h}, \\ &= \frac{\partial \mathcal{L}_n}{\partial p_n} \frac{\partial p_n}{\partial \mathbf{h}_n} \sum_{j=1}^n \left( \prod_{i=j+1}^n \frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}} \right) \frac{\partial \mathbf{h}_j}{\partial \mathbf{w}_h}.\end{aligned}$$

We can now substitute in some known values,

$$\frac{\partial \mathcal{L}_n}{\partial \mathbf{w}_h} = \frac{\partial \mathcal{L}_n}{\partial p_n} \odot \phi' \mathbf{h}_n^T \sum_{j=1}^n \left( \prod_{i=j+1}^n \phi' \mathbf{w}_{h,i}^T \right) \phi' \mathbf{h}_{j-1}^T. \quad (4.4)$$

Finally,  $\partial \mathcal{L}_n / \partial \mathbf{w}_x$  is derived in the same way as  $\partial \mathcal{L}_n / \partial \mathbf{w}_h$

$$\begin{aligned} \frac{\partial \mathcal{L}_n}{\partial \mathbf{w}_x} &= \frac{\partial \mathcal{L}_n}{\partial p_n} \frac{\partial p_n}{\partial \mathbf{h}_n} \sum_{j=1}^n \left( \prod_{i=j+1}^n \frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}} \right) \frac{\partial \mathbf{h}_j}{\partial \mathbf{w}_x}, \\ &= \frac{\partial \mathcal{L}_n}{\partial p_n} \odot \phi' \mathbf{h}_n^T \sum_{j=1}^n \left( \prod_{i=j+1}^n \phi' \mathbf{w}_{h,i}^T \right) \phi' \mathbf{x}_j^T. \end{aligned} \quad (4.5)$$

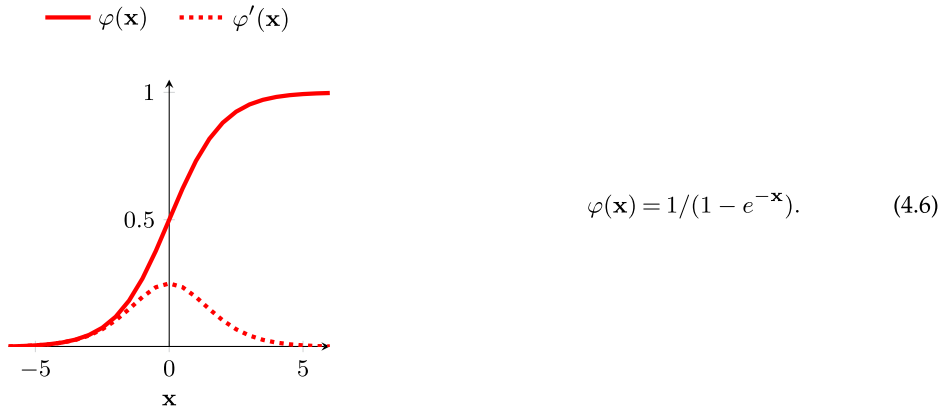
With  $\partial \mathcal{L}_n / \partial \mathbf{w}_p$ ,  $\partial \mathcal{L}_n / \partial \mathbf{w}_h$  and  $\partial \mathcal{L}_n / \partial \mathbf{w}_x$  in hand we can apply the same update rule shown in equation (2.5).

Aside from many-to-one encoding, RNNs can produce many predictions given many inputs, or act similarly to an MLP and produce one or many outputs given a single input. We will discuss the application of recurrent neural networks to astronomical data in §5, after we introduce gated recurrent neural networks.

### 4.3. Sidestepping the vanishing gradient problem

In the early 1990s, researchers identified a major issue with the training of deep neural networks through backpropagation. Hochreiter first formally examined the ‘vanishing gradient’ problem in their diploma thesis (Hochreiter [98], see also later work by Bengio *et al.* [99]). Owing to the vanishing gradient problem, it was widely believed that training very deep artificial neural networks from scratch via backpropagation was impossible. In this section, we will explore what the vanishing gradient problem is, and how contemporary end-to-end trained neural networks sidestep this issue.

First let us remind ourselves of the sigmoid activation function introduced in figure 6,



Equation (4.6) and its accompanying plot shows the output of a sigmoid function  $\varphi$  and its derivative  $\varphi'$ , when given an input  $\mathbf{x}$ .

Now, let us revisit the weight update rule for the  $(L - n)$ th layer of a feedforward MLP (equation (2.4))

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_{L-n}} = \frac{\partial \mathcal{L}}{\partial p_L} \odot \underbrace{\left( \prod_{i=1}^n \phi'_{L-i} \mathbf{w}_{L-i}^T \right)}_{\lim_{n \rightarrow \infty} \prod_{i=1}^n \phi'_{L-i} \mathbf{w}_{L-i}^T = 0} (\phi'_{L-n} \mathbf{p}_{L-n}^T). \quad (4.7)$$

If  $\varphi'$  is typically less than one (as in equation (4.6) and most other saturating nonlinearities) the product term in the above equation becomes an issue. In that case, we can see that the product rapidly goes to zero as  $n$  (the number of layers) becomes large.<sup>16</sup> If we study equation (4.4), we can see the same

<sup>16</sup>Likewise, if  $\varphi'$  is typically greater than one, the product term rapidly ‘explodes’ to infinity. This is known as the ‘exploding gradient’ problem, also first identified in Hochreiter [98].

problem also plagues RNNs as we backpropagate through hidden states

$$\frac{\partial \mathcal{L}_n}{\partial \mathbf{w}_n} = \frac{\partial \mathcal{L}_n}{\partial p_n} \odot \phi' \mathbf{h}_n^T \sum_{j=1}^n \underbrace{\left( \prod_{i=j+1}^n \phi' \mathbf{w}_{h,i}^T \right)}_{\lim_{n \rightarrow \infty} \prod_{i=j+1}^n \phi' \mathbf{w}_{h,i}^T = 0} \phi' \mathbf{h}_{j-1}^T. \tag{4.8}$$

Let us solidify this issue by reminding ourselves of equation (2.5)—the weight update rule for a network trained through backpropagation

$$\mathbf{w}_{\text{next}} = \mathbf{w} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{w}}. \tag{4.9}$$

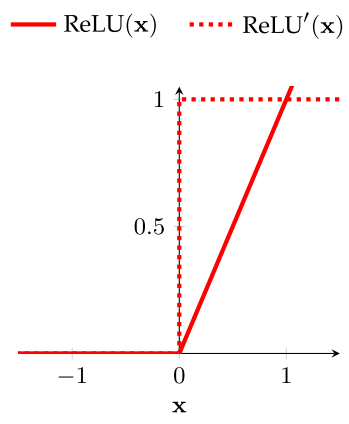
Combining equation (4.9) and the limits defined in equations (4.7) and (4.8) results in the below weight update rule in the limit  $n \rightarrow \infty$ .

$$\lim_{n \rightarrow \infty} \mathbf{w}_{\text{next}} = \mathbf{w}. \tag{4.10}$$

Equation (4.10) shows that learning via backpropagation slows as we move deeper into the network. This problem once again caused a loss of faith in the connectionist model, ushering in the second AI winter. It took until 2012 for a new boom to begin. In the following three subsections, we will explore some of the proposed partial solutions to the vanishing gradient problem and show how they came together to contribute to the current deep learning boom.

### 4.3.1. Non-saturating activation functions

We can see in equations (4.8) and (4.7) that if  $\phi' = 1$  then the product term does not automatically go to zero or infinity. If this is the case, why not simply design our activation function around this property? The rectified linear unit (ReLU; [46,47]) is an activation function that does precisely this,<sup>17</sup>



$$\text{ReLU}(\mathbf{x}) = \max(\mathbf{x}, 0). \tag{4.11}$$

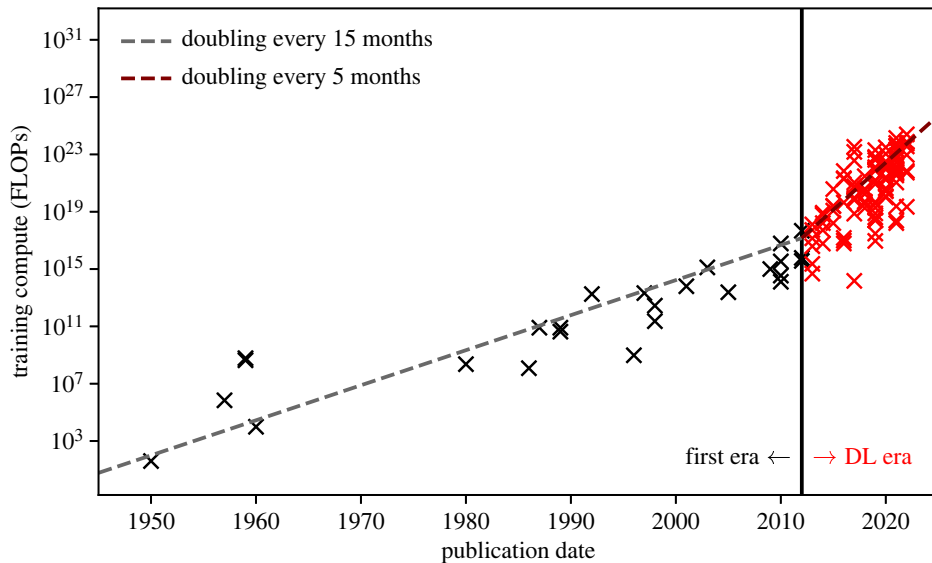
The gradient of ReLU is unity if the inputs are above zero, exactly the property we needed to mitigate the vanishing gradient problem. Similar non-saturating activation functions also share the ReLU gradient's useful property, see for example the exponential linear unit, Swish and Mish functions in figure 6.

### 4.3.2. Graphics processing unit acceleration

If we can speed up training, we can run an inefficient algorithm (such as backpropagation through saturating activations) to completion in less time. One way to speed up training is by using hardware that is specifically suited to the training of neural networks. Graphics processing units (GPUs) were originally developed to render video games and other intensive graphical processing tasks. These rendering tasks require a processor capable of massive parallelism. We have seen in the previous sections that neural networks trained through backpropagation also require many small weight update calculations. With this in mind, it is natural to try to accelerate deep neural networks using GPUs.

<sup>17</sup>ReLU is always zero if its inputs are less than 0, removing any signal for further training. This is known as the 'dying ReLU' problem, but is not as big of an issue as it first seems. Since contemporary deep neural networks are greatly overparametrized (see for example Frankle & Carbin [100] and other work on the 'lottery ticket hypothesis') backpropagation through the ReLU activation function can act as a pruning mechanism, creating sparse representations within the neural network and thus reducing training time even further [101].





**Figure 10.** If we plot the total number of floating point operations (FLOPs) required to train a neural network model, and compare it with the model's publication date, we can see a change in trend at around 2012. This corresponds to the popularization of GPU-accelerated training of very deep neural networks, with 2012 demarcating AI's 'Deep Learning Era' and the beginning of astronomy's second wave of connectionism (§5). Data are taken from Sevilla *et al.* [111].

In 2004, Oh & Jung [102] were the first to use GPUs to accelerate an MLP model, reporting a 20× performance increase on inference with an 'ATI RADEON 9700 PRO' GPU accelerated neural network. Shortly after, Steinkrau *et al.* [103] showed that backpropagation can also benefit from GPU acceleration, reporting a threefold performance increase in both training and inference. These two breakthroughs were followed by a flurry of activity in the area (e.g. [104–107]), culminating in a milestone victory for GPU accelerated neural networks at ImageNet 2012. AlexNet [108] won the ImageNet classification and localization challenges [109], scoring an unprecedented top-5 classification error of 16.4%, and a single object localization error of 34.2%. In both challenges, AlexNet scored over 10% better than the models in second place. Sutskever & Hinton's winning network was a CNN [46] trained through backpropagation [40,93], with ReLU activation [47] and dropout [110] as a regularizer.<sup>18</sup> The performance increase afforded by GPU-accelerated training enabled the network to be trained from scratch via backpropagation in a reasonable amount of time. The discovery that it is possible to train a neural network from scratch by using readily available hardware ultimately resulted in the end of connectionism's second winter, and ushered in the Cambrianesque deep learning explosion of the mid-to-late 2010s and the 2020s (figure 10).

### 4.3.3. Gated recurrent neural networks and residual networks

The long short-term memory unit (LSTM, [112,113])<sup>19</sup> mitigates the vanishing gradient problem by introducing a new hidden state, the 'cell state' ( $\mathbf{c}_n$ ), to the standard RNN architecture. This cell state allows the network to learn long-range dependencies, and we will show why this is the case via a brief derivation.<sup>20</sup> First, as always, let us study figure 11 and write down the forward pass equation for updating the cell state

$$\mathbf{c}_n = f(\mathbf{c}_{n-1}, \mathbf{h}_{n-1}, \mathbf{x}_n) + g(\mathbf{h}_{n-1}, \mathbf{x}_n),$$

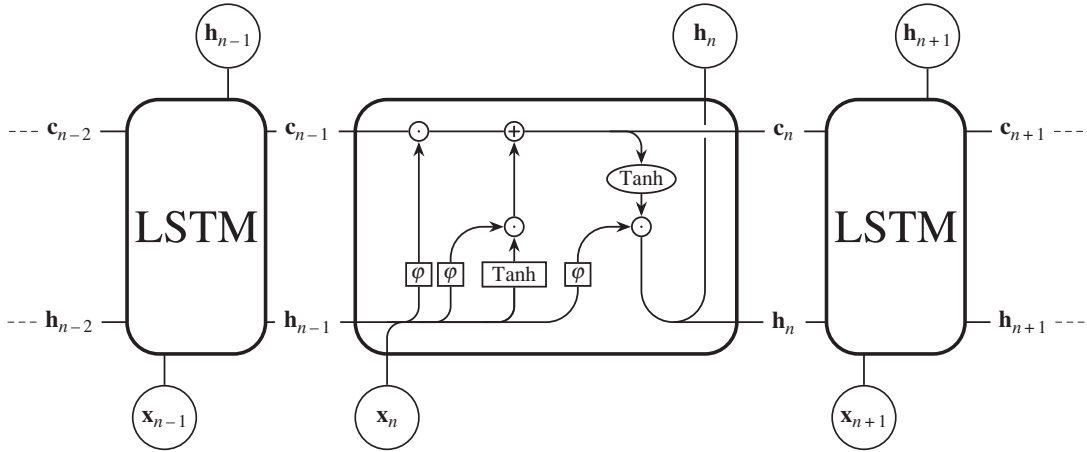
where  $f(\mathbf{c}_{n-1}, \mathbf{h}_{n-1}, \mathbf{x}_n) = \mathbf{c}_{n-1} \odot \varphi(\mathbf{h}_{n-1}, \mathbf{x}_n)$ . For brevity we define  $\varphi_n = \varphi(\mathbf{h}_{n-1}, \mathbf{x}_n)$ .

<sup>18</sup>Dropout reduces the amount of neural network overfitting—where a network performs well on the training set at the expense of performance on data it has not yet seen. One performs dropout by randomly removing a set of neurons at each training step, and using all neurons at test time. This set-up essentially trains a large ensemble of sub-models, whose average prediction outperforms that inferred by a single model.

<sup>19</sup>Compare also the gated recurrent unit (GRU, [114]).

<sup>20</sup>Here we loosely follow Bayer ([115], §1.3.4).





**Figure 11.** A set of sequential data  $\mathbf{x}_n$  is input into an LSTM network. Inside the cell  $\circ$  denotes elementwise operations and  $\square$  denotes neuronal layers.  $\varphi$  is the sigmoid activation function, and Tanh is the hyperbolic tangent activation function.  $\oplus$  is an elementwise addition,  $\odot$  is the Hadamard product, and line mergers are concatenations.  $\mathbf{c}_n$  is the cell state, and  $\mathbf{h}_n$  is the hidden state.

Like the RNN case (equations (4.4) and (4.5)), we will need to find  $\partial \mathbf{c}_n / \partial \mathbf{c}_{n-1}$  to calculate  $\nabla \mathcal{L}$ . Therefore,

$$\begin{aligned} \frac{\partial \mathbf{c}_n}{\partial \mathbf{c}_{n-1}} &= \frac{\partial f(\mathbf{c}_{n-1}, \mathbf{h}_{n-1}, \mathbf{x}_n)}{\partial \mathbf{c}_{n-1}} + \frac{\partial g(\mathbf{h}_{n-1}, \mathbf{x}_n)}{\partial \mathbf{c}_{n-1}} \rightarrow 0 \\ &= \frac{\partial \mathbf{c}_{n-1} \odot \varphi_n}{\partial \mathbf{c}_{n-1}}, \\ &= \mathbf{c}_{n-1} \frac{\partial \varphi_n}{\partial \mathbf{c}_{n-1}} \rightarrow 0 + \frac{\partial \mathbf{c}_{n-1}}{\partial \mathbf{c}_{n-1}} \varphi_n \rightarrow 1, \\ &= \varphi_n. \end{aligned}$$

Thus, if we want to backpropagate to a cell state deep in the network, we must calculate

$$\frac{\partial \mathbf{c}_n}{\partial \mathbf{c}_N} = \prod_{i=1}^{n-N} \varphi_i, \quad n > N. \quad (4.12)$$

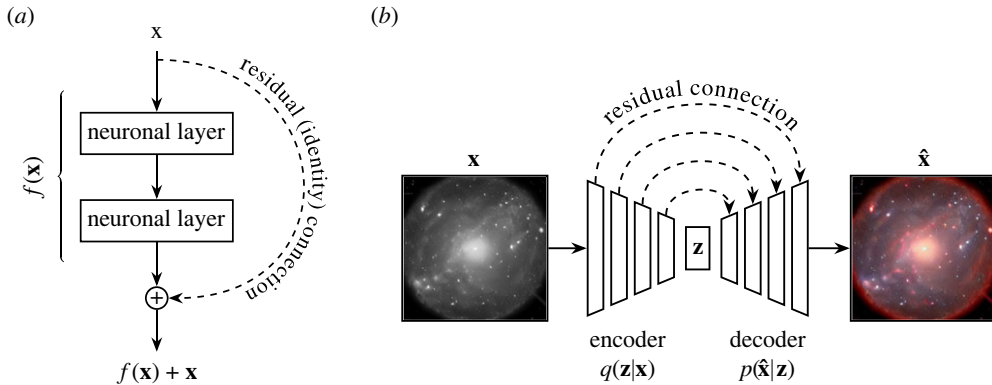
The product term above does not depend on the derivative of a saturating activation function, and so does not automatically vanish as  $N$  goes to  $\infty$ . This means that a gradient signal can be carried through the LSTM cell state without losing amplitude and vanishing.<sup>21</sup>

We can use a technique derived from the LSTM to solve our vanishing gradient problem for deep feedforward neural networks (as studied in §2.2). Srivastava *et al.* [118] do this by applying the concept of the LSTM's cell state to their deep convolutional 'highway network'. The highway network uses gated connections to modulate the gradient flow back through neuronal layers. Later work by He *et al.* [119] introduces the residual network (ResNet) by taking a highway network and simplifying its connections. They apply an elementwise addition (or 'residual connection') in place of the highway network's gated connection (figure 12a). One can go even further with residual connections, as Ronneberger *et al.* [120] demonstrate with their U-Net model. The U-Net combines residual connections with an autoencoder-like architecture (figure 12b). The U-Net has gone on to become the *de facto* network for many tasks that require an input and output of the same size (such as segmentation, colourization and style transfer).

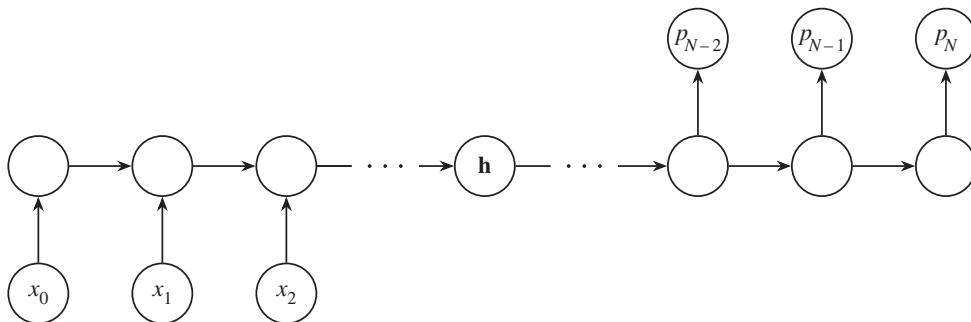
#### 4.4. Translation, attention and transformers

Theoretically, gated RNNs (GRNNs) such as the LSTM can learn very long-range dependencies (see equation (4.12) and its accompanying text). In practice, GRNNs tend to forget information about distant inputs. This is because the GRNN lacks unmediated access to inputs beyond the immediate antecedent as a consequence of its recurrent architecture. The problem is especially apparent in neural machine

<sup>21</sup>Which is great in theory. In practice, LSTMs still have trouble learning very long-range dependencies due to their reliance on recurrent processing [116]. Transformer networks [117] are an architecture that uses the concept of attention to address this issue. We will discuss transformer networks in §4.4.



**Figure 12.** Panel (a) shows the residual connection as originally introduced in He *et al.* [119]. Panel (b) shows an application of the residual connection to an autoencoder-like U-Net architecture [120], in this case colourizing an astronomical object. Here,  $\mathbf{z}$  is a compressed shared representation of  $\mathbf{x}$  and  $\hat{\mathbf{x}}$ .



**Figure 13.** A sequence to sequence (Seq2Seq; [116]) model. A sequence  $\mathbf{x}$  is input into a GRNN. The final hidden state ( $\mathbf{h}$ ) of the input network is then passed into a second GRNN. The second GRNN then unrolls to predict an output sequence  $\mathbf{p}$ . Owing to the hidden state acting as an intermediary,  $\mathbf{x}$  and  $\mathbf{p}$  need not be of equal length.

translation tasks that require knowledge of an entire sequence to produce an output, such as language to language translation. Figure 13 shows such a sequence to sequence (Seq2Seq; [116]) model. Seq2Seq translates between two sets of sequential data by sharing a hidden state between two GRNN units. In figure 13, we can see that the shared information is bottlenecked by the hidden state. Therefore, to resolve the GRNN ‘forgetting problem’ we must find a way to avoid any recursion, or serial processing of input and output. We can do this by providing the neural network access to all input while it is calculating an output. This was the primary motivation behind the transformer architecture [117,121].

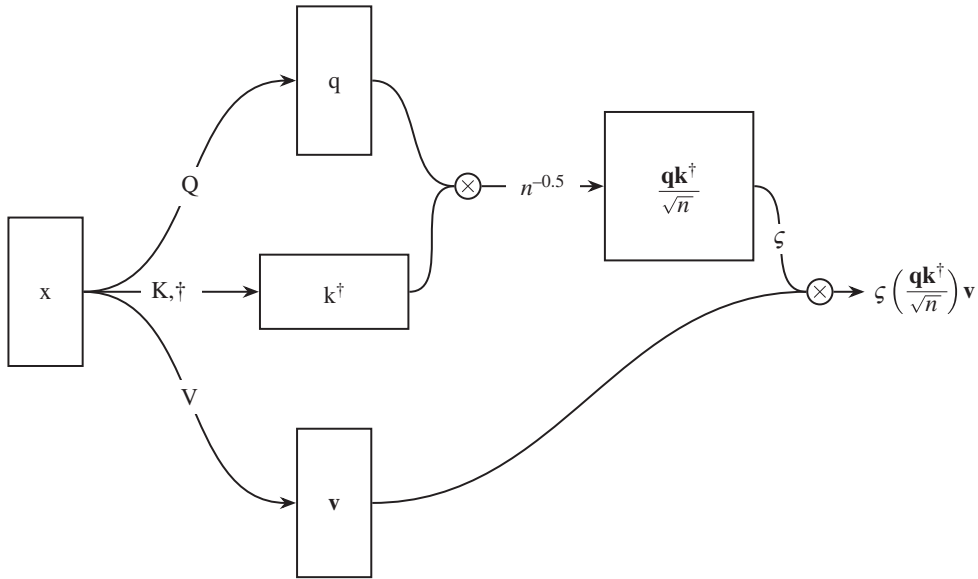
Modern transformer architectures consist of a series of self-attention layers interspersed with other layer types.<sup>22</sup> Self-attention as described in Vaswani *et al.* [117] is shown in figure 14. Intuitively, it captures the relationships between quanta within a data input. To perform self-attention, we first take an input sequence

$$\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_n],$$

where  $\mathbf{x}$  can be any sequence, such as a sentence, a variable star’s time series, or an unravelled galaxy image.<sup>23</sup> This sequence has a maximum length ( $n$ ) that must be defined at train time, but we can process shorter sequences by masking out any surplus values so that they do not affect the loss. Here we will follow the literature and refer to  $[x_1, \dots, x_n]$  as tokens. As we can see in figure 14, the input is passed through a trainable pair of weight matrices  $\mathbf{Q}$  (or ‘query’) and  $\mathbf{K}$  (or ‘key’). The output matrices  $\mathbf{q}$  and  $\mathbf{k}^\dagger$

<sup>22</sup>In the original transformer formulation described in Vaswani *et al.* [117], the network consisted of a connected ‘encoder’ and ‘decoder’ section much like a Seq2Seq model (figure 13). Later work has found this to be an unnecessary complication. For example, the generative pretrained transformer (GPT) 2 and 3 models [17,122] consist of only decoder layers, and the bidirectional encoder representations from transformers (BERT) model consists of only encoder layers [123].

<sup>23</sup>One can go very general with this, as DeepMind demonstrated with their ‘Gato’ transformer model [124]. Gato can predict sequences for myriad tasks, from operating a physical robotic arm, to completing natural language sentences, to playing Atari games.



**Figure 14.** An input ( $\mathbf{x}$ ) is fed into a self-attention mechanism. The weights used to produce the query ( $\mathbf{q}$ ), key ( $\mathbf{k}$ ) and value ( $\mathbf{v}$ ) matrices are learnt via backpropagation. Here the learnt weights are denoted as the capitalized versions of their child matrices.  $\mathbf{q}$  and  $\mathbf{k}$  are normalized and multiplied together, and a softmax nonlinearity ( $\varsigma$ ) is applied. Finally,  $\mathbf{v}$  is multiplied with output of the upper path and the final output is fed forward to the next neuronal layer.  $\otimes$  denotes a matrix multiplication.

are then multiplied together to yield

$$(\mathbf{Q} \cdot \mathbf{x})(\mathbf{K} \cdot \mathbf{x})^\dagger = \mathbf{qk}^\dagger = \begin{bmatrix} Q_1x_1K_1x_1 & Q_1x_1K_2x_2 & \cdots & Q_1x_1K_nx_n \\ Q_2x_2K_1x_1 & Q_2x_2K_2x_2 & \cdots & Q_2x_2K_nx_n \\ \vdots & \vdots & \ddots & \vdots \\ Q_nx_nK_1x_1 & Q_nx_nK_2x_2 & \cdots & Q_nx_nK_nx_n \end{bmatrix}. \tag{4.13}$$

We can see that equation (4.13) describes the relationships between tokens within  $\mathbf{x}$ . For example, if  $x_1$  is similar semantically to  $x_2$ , we would expect  $Q_1x_1K_2x_2$  and  $Q_2x_2K_1x_1$  to have a high value. We then normalize  $\mathbf{qk}^\dagger$  to mitigate vanishing gradients (see footnote 16) and apply a softmax nonlinearity so that the maximum weighting (or similarity) is one and the similarity values sum to unity.

Meanwhile, the input sequence  $\mathbf{x}$  is passed through the neuronal layer  $\mathbf{V}$ , resulting in a weighted representation  $\mathbf{v}$ ,

$$\mathbf{V} \cdot \mathbf{x} = \mathbf{v} = [V_1x_1 \quad V_2x_2 \quad \cdots \quad V_nx_n].$$

$\mathbf{v}$  is multiplied with the similarity matrix  $\varsigma(\mathbf{qk}^\dagger/\sqrt{n})$ . This process weighs similar tokens within the sequence higher, increasing their relative importance in later neuronal layers.

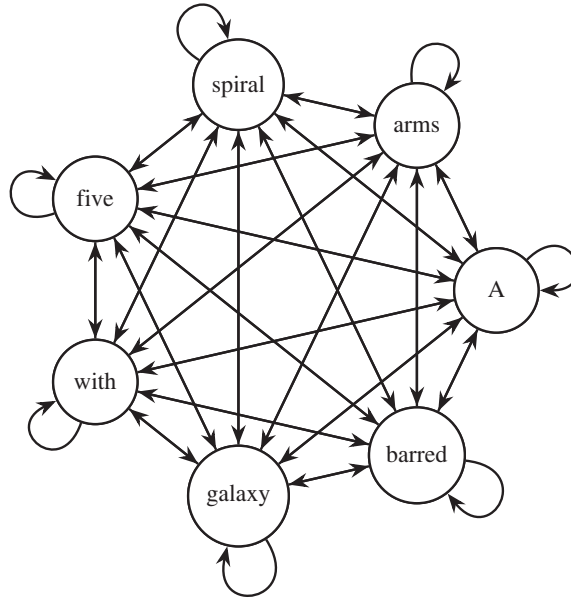
We will use an astronomical example to solidify our understanding of the self-attention mechanism. Let us assume that our self-attention mechanism is attending to a natural language caption describing a galaxy’s morphology that has been provided by a citizen scientist. The caption could be something like:

$\mathbf{x} = A \text{ barred galaxy with five spiral arms,}$

with each word acting as a separate token. Let us imagine that we put this prompt into our self-attention mechanism,

$$(\mathbf{Q} \cdot \mathbf{x})(\mathbf{K} \cdot \mathbf{x})^\dagger = \mathbf{qk}^\dagger = \begin{matrix} & \begin{matrix} A & \text{barred} & \text{galaxy} & \text{with} & \text{five} & \text{spiral} & \text{arms} \end{matrix} \\ \begin{matrix} A \\ \text{barred} \\ \text{galaxy} \\ \text{with} \\ \text{five} \\ \text{spiral} \\ \text{arms} \end{matrix} & \begin{bmatrix} 0.7 & 0.2 & 0.1 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.2 & 0.5 & 0.3 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.1 & 0.3 & 0.5 & 0.1 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.1 & 0.8 & 0.1 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.1 & 0.6 & 0.3 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.3 & 0.5 & 0.2 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.8 \end{bmatrix} \end{matrix}.$$

We can see that in the above matrix higher values have been assigned to pairs of words that are more closely related within the sentence. For example, the weight between ‘barred’ and ‘galaxy’ is relatively high (0.3), as the term ‘barred’ describes a feature of galaxy. Similarly, the weight between ‘five’ and



**Figure 15.** We can think of  $\mathbf{qk}^\dagger$  within self-attention as a graph of relationships between a prompt and itself. Each of the edges in this graph represents the weight shared between a pair of tokens in the input sequence.

'spiral' is also high (0.3), as these words together define the number of spiral arms in the galaxy. Conversely, lower weights have been assigned to word pairs that are less related, such as 'A' and 'with' (0.0). As shown in figure 15, one can think of these relationships between tokens within our sequence as a learnt mathematical graph.<sup>24</sup> Now that we have calculated  $\mathbf{qk}^\dagger$ , we can use this matrix to weigh our example sentence as shown in figure 14. This weighting gives the subsequent layers in our neural network an awareness of the relationships between the tokens in our sequence.

## 5. Astronomy's second wave of connectionism

Compared with classical connectionist approaches<sup>25</sup> deep learning as outlined in §4 does not require an extraction of emergent parameters to train its models. CNNs in particular are well suited to observing raw information within image-based data. Likewise, RNNs are well suited to observing the full raw information within a time series. Astronomy is rich with both types of data, and in this section we will review the history of the application of CNN, RNN and transformer models to astronomical data.

### 5.1. Convolutional neural network applications

It did not take long after Krizhevsky *et al.* [108] established CNNs as the de facto image classification network for astronomers to take notice: in 2014, they were applied in the search for pulsars [129] as part of an ensemble of methods. Zhu *et al.* [129] found that their ensemble was highly effective, with 100% of their test set pulsar candidates being ranked within the top 961 of the 90 008 test candidates. Shortly after, Hála [130] described the use of one-dimensional CNNs for a ternary classification problem. They found that their model is capable of classifying one-dimensional spectra into quasars, galaxies and stars to an impressive accuracy. CNNs have also been extensively used in galaxy morphological classification. First on the scene was Dieleman *et al.* [131]. They used CNNs to classify galaxy morphology parameters as defined in the Galaxy Zoo dataset [132] from galaxy imagery. They observed their galaxies via the SDSS, and found a 99% consensus between the Galaxy Zoo labels, and the CNN classifications. Huertas-Company *et al.* [133] showed that the CNN introduced in Dieleman *et al.* [131] is equally applicable to the morphological classification of galaxies in the CANDELS fields [134]. Likewise, Aniyán & Thorat [135] showed that CNNs are capable of classifying radio galaxies. The combined work of Dieleman *et al.* [131], Huertas-Company *et al.* [133] and Aniyán

<sup>24</sup>This view demonstrates that transformers can be thought of as a class of graph neural network—a network that is tasked with learning the relationships between nodes in a graph. One can also approach this task with a feed forward neural network (§2.2; [125]), convolutional architecture (§4.1; [126,127]) or with a recurrent architecture (§4.3.3; [128]).

<sup>25</sup>This includes most MLP applications in astronomy, see §3.

& Thorat [135] confirms that CNNs are equally applicable to visually dissimilar surveys, with little-to-no modification. Looking a little further afield, Wilde *et al.* [136] used a deep CNN model to classify simulated lensing events. They also applied some interpretability techniques to their data, using occlusion mapping [137], gradient class activation mapping [138] and Google's DeepDream to prove that the CNN was indeed classifying via observing the gravitational lenses. Alternative CNN models have also been used, such as the U-Net (figure 12*b*). The U-Net was initially developed to segment biological imagery [120]. Its first use in astronomy was related: Akeret *et al.* [139] used a U-Net [120] CNN to isolate via segmentation, and ultimately remove, radio frequency interference from radio telescope data. Likewise, Berger & Stein [140] used a three-dimensional U-Net (V-Net; [141]) to predict and segment out galaxy dark matter haloes in simulations, and Aragon-Calvo [142] used a V-Net to segment out the cosmological filaments and walls that make up the large-scale structure of the Universe. Hausen & Robertson [143] demonstrate that a U-Net is capable of performing pixelwise semantic classification of objects in HST/CANDELS imagery, thus proving that U-Nets are capable of useful work directly within large imaging surveys, particularly in the deblending of overlapping objects, which is a perennial challenge in deep imaging. The U-Net in Lauritsen *et al.* [144] is used to super-resolve simulated submillimetre observations. They found that the U-Net could successfully do this when using a loss comprising the L1 loss and a custom loss that measures the distance between predicted and ground truth point sources. Choma *et al.* [145] were the first to demonstrate that graph convolutional neural networks (GCNNs) are useful within astronomical context. They showed that their three-dimensional GCNN could classify signals from the IceCube neutrino observatory, and found that it outperformed both a classical method, and a standard three-dimensional CNN. Villanueva-Domingo *et al.* [146,147] demonstrated that EdgeNet—a class of GCNN—can estimate halo masses when given the positions, velocities, stellar masses and radii of the host galaxies [148]. The authors also demonstrated that EdgeNet can estimate the halo masses of both Andromeda and the Milky Way. We must conclude from the studies described in this subsection that CNNs are effective classifiers and regressors of image-based astronomical data.

## 5.2. Recurrent neural network applications

RNNs were first applied in astronomy very close to home; Aussem *et al.* [149] predicted atmospheric seeing for observations from the European Southern Observatory's Very Large Telescope, and the prediction of geomagnetic storms given data on the solar wind was also explored in the mid-to-late 1990s and early 2000s ([150,151] and other work from the same group; [152]).

The first use of RNNs for classification in astronomy was carried out in a prescient study by Brodrick *et al.* [153]. They describe the use of an RNN-like Elman network [154]. Their RNN was tasked with the search for artificially generated narrowband radio signals that resemble those that may be produced by an extraterrestrial civilization. They found that their model had a test set accuracy of 92%, suggesting that RNNs could be a useful tool in the search for extraterrestrial intelligence. More than a decade after Brodrick *et al.* [153], Charnock & Moss [155] used an LSTM (figure 11) to classify simulated supernovae. They describe two classification problems. One, a binary classification between type-Ia and non-type-Ia supernovae, and the other a classification between supernovae types I, II and III. For their best performing model, they report an accuracy of more than 95% for their binary classification problem, and an accuracy of over 90% for their trinary classification. This study cemented the usefulness of RNNs for classification problems in astronomy. Charnock & Moss [155] were followed by numerous projects studying the use of RNNs for classification of time-series astronomical data. A non-exhaustive list of modern RNN use in astronomy includes: stochastically sampled variable star classification [156], exoplanet instance segmentation [157], variable star/galaxy sequential imagery classification [158] and gamma ray source classification [159]. We must conclude from these studies that RNNs are effective classifiers of astronomical time series, provided that sufficient data are available.

Of course, recurrent networks are not limited to classification; they can also be used for regression problems. First, Weddell & Webb [160] successfully used an echo state network [161] to predict the point spread function of a target object in a wide field of view. Capizzi *et al.* [162] used an RNN to inpaint missing NASA Kepler time series data for stellar objects. They found that their model could recreate the missing time series to an excellent accuracy, suggesting that the RNN could internalize information about the star it was trained on. As in the classification case, research into the use of RNNs for regression problems picked up massively in the late 2010s, and here we will highlight a selection of these studies that represent the range of RNN use cases. Shen *et al.* [163] used both an LSTM and an autoencoder-based RNN to denoise gravitational wave data, and Morningstar *et al.*

[164] used a recurrent inference machine to reconstruct gravitationally lensed galaxies. Liu *et al.* [165] used an LSTM to predict solar flare activity. From these studies, similarly to the classification case above, we can once again conclude that RNNs are effective regressors of astronomical time series.

RNNs have also been used in cases that are a little more unconventional. For example, Kügler *et al.* [166] used an autoencoding RNN (specifically an echo state network) to extract representation embeddings of variable main sequence stars. They find that these embeddings capture some emergent properties of these variable stars, such as temperature and surface gravity, suggesting that clustering within the embedding space could result in semantically meaningful variable star classification. We will revisit this line of research when we explore representation learning within astronomy in detail in §8. An example of more drastic cross-pollination between ideas within deep learning and those within astronomy is Smith *et al.* [167]. They use an encoder–decoder network comprising a CNN encoder and RNN decoder to predict surface brightness profiles of galaxies. This class of neural network was previously used extensively within natural language image captioning, and by treating surface brightness profiles as ‘captions’ their model was capable of prediction over 100× faster than the previous classical, human-agent-based method.

### 5.3. Transformer applications

Although initially used for natural language, transformers have also been adapted for use in imagery, first by Parmar *et al.* [168], and also in Dosovitskiy *et al.* [18]. To the best of our knowledge, transformers have not yet been applied to astronomical imagery, but they have started to find use in time-series astronomy. Donoso-Oliva *et al.* [169] used BERT [123] to generate a representation space for light curves in a self-supervised manner. Morvan *et al.* [170] used an encoding transformer to denoise light curves from the Transiting Exoplanet Survey Satellite (TESS, [171]) and show that the denoising surrogate task results in an expressive embedding space. Pan *et al.* [172] also use a transformer model to analyse light curves for exoplanets. Transformers have taken the fields of natural language processing and computer vision by storm (§9), and so if we extrapolate from trends in other fields we expect to see many more examples of transformers applied to astronomical use cases in the near future. We will revisit the transformer architecture in the context of foundation models ([173] and references therein) and their possible future astronomical applications in §9.

### 5.4. A problem with supervised learning

Supervised learning requires a high-quality labelled dataset to train a neural network. In turn, these datasets require laborious human intervention to create, and so supervised data is in short supply. One can avoid this issue by prompting the deep learning model to gather semantic information from entirely unlabelled data. This learnt semantic information can then be accessed through a hidden descriptive ‘latent space’, and then used for downstream tasks like data generation, classification and regression. Indeed, all of the networks described previously in this review can be repurposed for non-supervised tasks, and in §§6 and 7 we will explore some deep learning frameworks that do not require supervision.

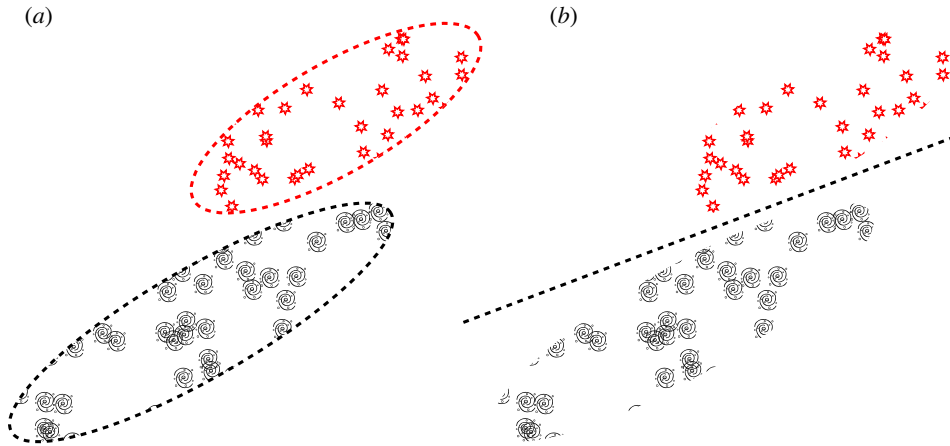
## 6. Deep generative modelling

In this section, we discuss generative modelling within the context of astronomy. Unlike discriminative models, generative models explicitly learn the distribution of classes in a dataset (figure 16). Once we learn the distribution of data, we can use that knowledge to generate new synthetic data that resembles that found in the training dataset. In the following subsections, we will explore in detail three popular forms of deep generative model: the variational autoencoder (§6.1), the generative adversarial network (§6.2) and the family of score-based (or diffusion) models (§6.3). Finally, in §8 we discuss applications of deep generative modelling in astronomy.

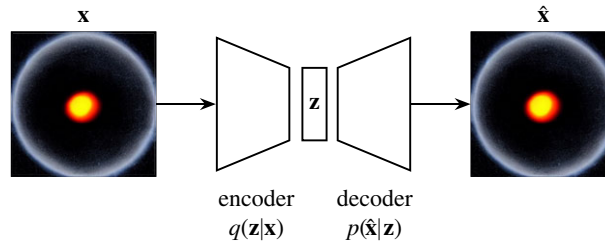
### 6.1. (Variational) autoencoders

Autoencoders have long been a neural network architectural staple. In a sister paper to backpropagation’s popularizer, Rumelhart *et al.* [174] demonstrate backpropagation within an autoencoder. Figure 17 demonstrates the basic neural network autoencoder architecture. An autoencoder is tasked with recreating some input data, squeezing the input information ( $\mathbf{x}$ ) into a bottleneck latent vector ( $\mathbf{z}$ ) via a





**Figure 16.** Here we show a possible latent space representation of a set of galaxies and a set of stars. A latent (or embedding) space is a compressed representation of a set of objects where similar objects are clustered closer together than dissimilar objects. While this space is often highly dimensional, here we project our latent space onto two dimensions for visualization purposes. In (a), we see a generative model attempting to learn the probability distributions of the latent representation of a dataset that contains a set of galaxies and a set of stars. In (b), we see a discriminative model attempting to learn the boundary that separates the star and galaxy types.



**Figure 17.** An autoencoder [174] attends to an image of a black hole.  $z$  is a latent vector and  $x$  is a sample from a training set. The encoder,  $q$  learns to encode the incoming data into a latent vector while the decoder  $p$  takes as input  $z$  and attempts to recreate  $x$ .

neural network  $q(z|x)$ .  $z$  is then expanded to an imitation of the input data ( $\hat{x}$ ) by a second neural network  $p(\hat{x}|z)$ . The standard autoencoder is trained via a reconstruction loss;  $\mathcal{L}_R(x, \hat{x})$ , where  $\mathcal{L}_R(x, \hat{x})$  measures the difference in pixelspace between  $x$  and  $\hat{x}$ .

Naively, one would think that once trained, one could ‘just’ sample a new latent vector, and produce novel imagery via the decoding neural network  $p(\hat{x}|z)$ . We cannot do this, as autoencoders trained purely via a reconstruction loss have no incentive to produce a smoothly interpolatable latent space. This means we can use a standard autoencoder to embed and retrieve data contained in the training set, but cannot use one to generate new data. To generate new data we require a smooth latent space, which variational autoencoders (VAEs, figure 18) produce by design [175].

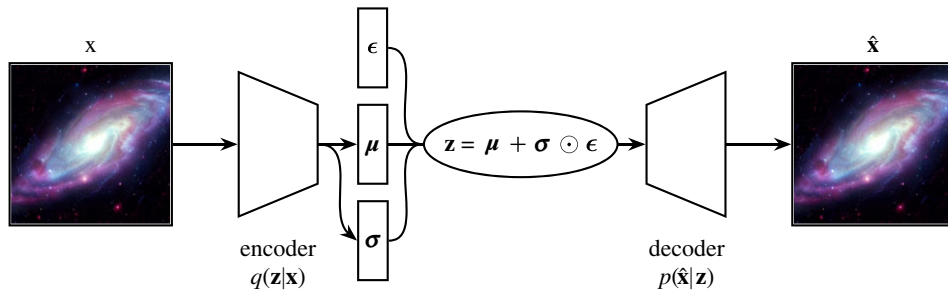
A VAE differs from the standard autoencoder by enforcing a spread in each training set samples’ latent vector. We can see in figure 18 how this is done; instead of directly predicting  $z$  the encoder  $q$  predicts two vectors,  $\mu$  and  $\sigma$ .  $z$  is then sampled stochastically via the equation

$$z = \mu + \sigma \odot \epsilon, \quad (6.1)$$

where  $\odot$  is the Hadamard product, and  $\epsilon$  is noise generated externally to the neural network graph.<sup>26</sup> This spread results in similar samples overlapping within the latent space, and therefore we end up with a smooth latent space that we can interpolate through. However, currently there is no incentive for the neural network to provide a coherent, compact global structure in the latent space. For that we require a regularization term in the loss. This regularization is provided via the Kullback–Leibler (KL) divergence, which is a measure of the difference between two probability distributions. A standard VAE uses the KL divergence to push the latent distribution towards the standard normal distribution, incentivizing a compact, continuous latent space. Hence, the final VAE loss is a combination of the

<sup>26</sup>To avoid breaking the backpropagation chain the VAE injects noise via an external parameter,  $\epsilon$ . This is described in Kingma & Welling [175] as the ‘reparametrization trick’.





**Figure 18.** A variational autoencoder [175] operates on a spiral galaxy.  $\mathbf{z}$  is a latent vector and  $\mathbf{x}$  is a sample from the training set. The encoder,  $q$  learns to compress the incoming data into a latent vector that encodes the normal distribution. The decoder  $p$  takes as input  $\mathbf{z}$  and attempts to recreate  $\mathbf{x}$ .

reconstruction loss and KL divergence:

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_R(\mathbf{x}, \hat{\mathbf{x}}) + \text{KL}(q(\mathbf{z}|\mathbf{x})\|\rho), \quad (6.2)$$

where  $\rho$  is some prior. In a standard VAE  $\rho = \mathcal{N}(\mathbf{0}, \mathbb{1})$ .

In practice, VAEs are able to generate smooth and coherent samples, as they model the data distribution explicitly, which also means that we can perform latent space arithmetic on the latent vector—such as interpolation, reconstruction and anomaly detection [175]. Their explicit learning of the latent vector ( $\mathbf{z}$ ) means that they can trivially be repurposed for semi-supervised, self-supervised and supervised downstream tasks by manipulating  $\mathbf{z}$  [176,177]. However, the quality of samples generated by VAEs is lower than that of generative adversarial networks or score-based generative models [178]. This reduction in quality is due to the VAE’s simple posterior  $q(\mathbf{z}|\mathbf{x})$ , but one can mitigate this shortcoming by iteratively approaching a more complex posterior.<sup>27</sup> To regularize the latent space, VAEs require an assumption of the prior distribution which requires some knowledge of the dataset, although often this can be set as ‘just’ a normal distribution as shown in equation (6.2).

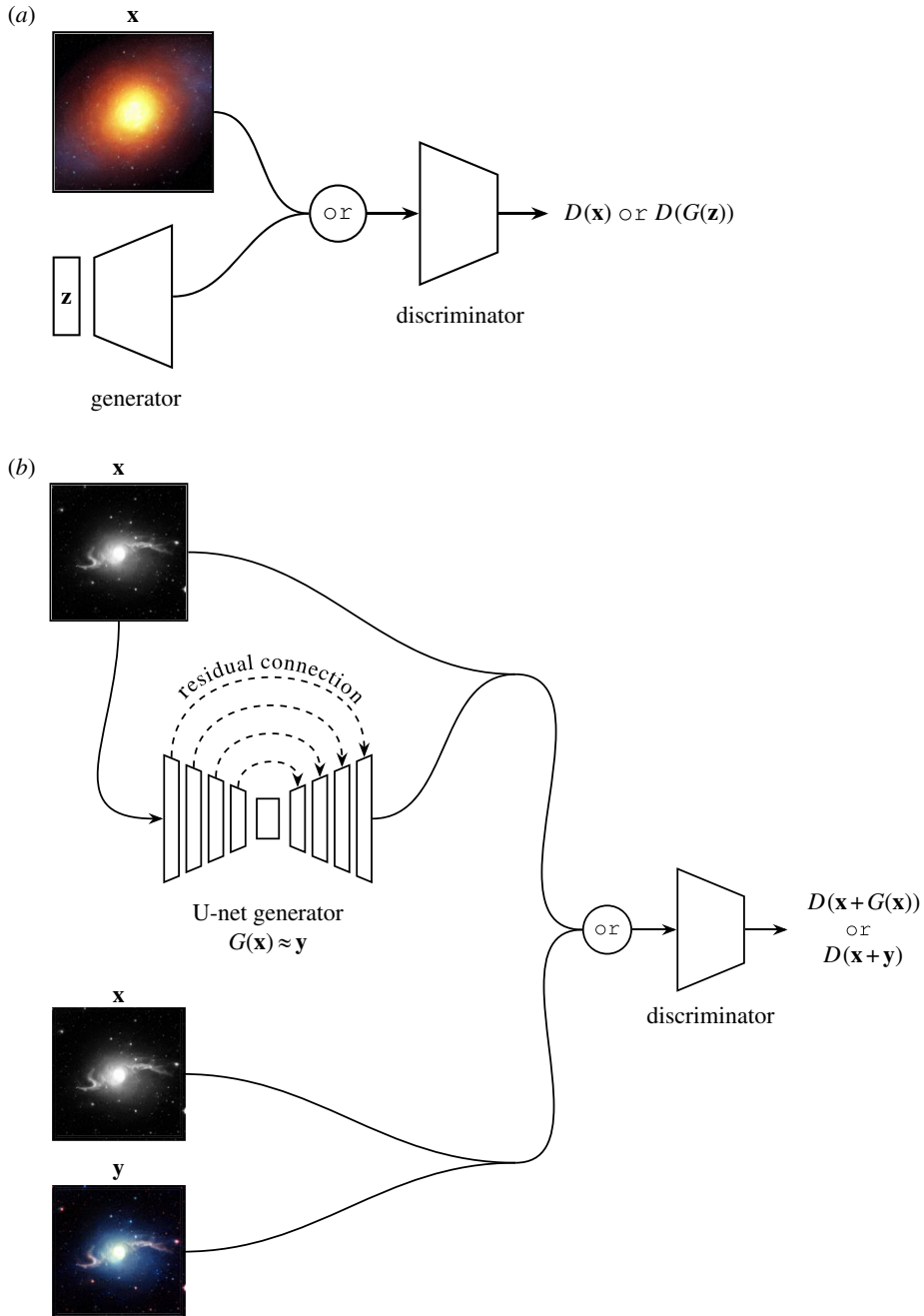
## 6.2. Generative adversarial networks

Generative adversarial networks (GAN, [183]) can be thought of as a minimax game between two competing neural networks. If we anthropomorphize, we can gain an intuition for how a GAN learns: let us imagine an art forger and an art critic. The forger wants to paint paintings that are similar to famous expensive works, and needs to fool the critic when selling these paintings. Meanwhile, the critic wants to ensure that no reproductions are sold, and so they need to accurately determine whether any painting is an original or a reproduction. At first, our forger is a poor painter, and so the critic can easily identify our forger’s works. However, the forger learns from the critic’s choices and produces more realistic paintings. As the forger’s paintings improve, the critic also learns better methods for detecting forgeries. This minimax game incentivizes the critic to keep improving their classifications, and the forger to keep improving their painting. If this continues, we get to a point where the forger’s works are indiscernible from the real thing—the forger has learnt to perfectly mimic the dataset! In a GAN, we name the critic the discriminator ( $D$ ), and we name the forger the generator ( $G$ ).

In Goodfellow *et al.*’s original GAN formulation (figure 19*a*),  $G$  and  $D$  are neural networks (typically CNNs, although other architectures can be used) that compete during training in a minimax game where  $G$  aims to maximize the probability of  $D$  mispredicting that a generated datapoint is sampled from the real dataset [183].  $G$  takes as input a randomly sampled latent vector  $\mathbf{z}$ , and outputs a synthetic datapoint  $G(\mathbf{z})$ .  $D$  takes either this synthetic datapoint, or a real datapoint  $\mathbf{x}$ , and outputs  $D(G(\mathbf{z}))$  or  $D(\mathbf{x})$ . This output is the probability that the datapoint is drawn from the real dataset. To train the network, we can write the GAN adversarial loss like so

$$\mathcal{L}_D = -(\mathbb{E}_{\mathbf{x}}[\log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z})))])$$

<sup>27</sup>Interestingly, this iterative approximation is similar to the approach used in the training of score-based generative models and diffusion models [179], and the similarities between the training methods of state of the art in VAE models and SBGMs are striking. For example, the Vector-Quantized VAE, Very Deep VAE and the Nouveau VAE all use a hierarchical architecture that iteratively injects latent codes that are used to produce finer and finer detail in the generated image [180–182].



**Figure 19.** The GAN and Pix2Pix models. (a) A typical GAN according to Goodfellow *et al.* [183].  $\mathbf{z}$  is a noise vector, and  $\mathbf{x}$  is a sample from the training set. The discriminator learns to classify the incoming images as either fake or real, and the generator learns to fool the discriminator by producing realistic fakes. (b) A Pix2Pix-like model with a U-Net generator [120,184]. The discriminator learns to classify the incoming image tuples as either fake or real. Meanwhile, the generator learns to fool the discriminator by approximating the colourization function mapping  $\mathbf{x} \rightarrow \mathbf{y}$ . Line mergers denote channel-wise concatenations.

and

$$\mathcal{L}_G = \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z})))]$$

where here we attempt to minimize both  $\mathcal{L}_D$  and  $\mathcal{L}_G$ . In practice, we train the networks by alternating freezing the weights of  $G$  and backpropagating  $\mathcal{L}_D$ , and then freezing the weights of  $D$  and backpropagating  $\mathcal{L}_G$  for each training batch. In this way, the networks' weights are updated to follow  $\nabla_{\mathbf{w}}\mathcal{L}_G$  and  $\nabla_{\mathbf{w}}\mathcal{L}_D$  downwards until the distribution of  $G(\mathbf{z})$  closely resembles that of the real dataset. Once trained,  $G$  can be used to generate entirely novel synthetic data that closely resembles (but is not identical to) the training set data.

One can condition a GAN to guide the network towards a desired output image [185]. To do this, we alter the adversarial loss so that it is conditioned on a label  $\mathbf{y}$

$$\mathcal{L}_D = -(\mathbb{E}_{\mathbf{x}}[\log(D(\mathbf{x}|\mathbf{y}))] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}|\mathbf{y})))])$$

and

$$\mathcal{L}_G = \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}|\mathbf{y})))].$$

As an example, if we set  $\mathbf{y}$  as the redshift of the galaxies in the training set, we could use a conditional GAN to guide the network to generate galaxies of a certain redshift. Furthermore, we are not restricted to conditioning single values; GANs can also be conditioned on entire images. In figure 19*b*, we see that the GAN adversarial loss can be used to translate between image domains [184]. In Isola *et al.*'s Pix2Pix model, the generator takes as input an image  $\mathbf{x}$ , and attempts to produce a related image  $\mathbf{y}$ . Meanwhile, the discriminator attempts to discern whether the  $(\mathbf{x}, \mathbf{y})$  pair that it is given is sampled from the training set, or the generator. Otherwise, Pix2Pix is trained in the same way as the standard GAN.

GANs are capable of generating high-quality, sharp and realistic samples [186,187]. They have long been a sweetheart of the deep generative learning community, having been used for various state-of-the-art applications, such as data embedding (e.g. [188]), style transfer (e.g. [189]), super-resolution (e.g. [190]), and image inpainting and object removal (e.g. [191]). Unfortunately, however, GANs have some downsides. They are quite difficult to train; maintaining the balance between the generator and discriminator networks is challenging and requires careful fine-tuning [192].  $G$  and  $D$  must work in tandem and one cannot overpower the other or learning will cease. One of the most famous symptoms of this imbalance is mode collapse, where  $G$  only generates a limited variety of samples that reliably fool  $D$ . This instability during training makes it quite a time-consuming task to find a stable network architecture if one is designing a GAN themselves. Finally, the GAN adversarial losses are relative and so are not representative of the image quality. This is not the case for the VAE and score-based generative model (SBGM) families of models.

### 6.3. Score-based generative modelling and diffusion models

Diffusion models were introduced by Sohl-Dickstein *et al.* [193] and were first shown to be capable of producing high-quality synthetic samples by Ho *et al.* [194]. Diffusion models are part of a family of generative deep learning models that employ denoising score matching via annealed Langevin dynamic sampling (first explored by Hyvärinen [195] and Vincent [196]. More recent work can be found in [194,197–200]). This family of SBGMs can generate imagery of a quality and diversity surpassing state-of-the-art GAN models [183], a startling result considering the historic disparity in interest and development between the two techniques [200–203]. SBGMs can super-resolve images [204,205], translate between image domains [206], separate superimposed images [207] and in-paint information [200,204].

Diffusion models define a diffusion process that projects a complex image domain space onto a simple domain space. In the original formulation, this diffusion process is fixed to a predefined Markov chain  $q(\mathbf{x}_t | \mathbf{x}_{t-1})$  that adds a small amount of Gaussian noise with each step. As figure 20 shows, this 'simple domain space' can be noise sampled from a Gaussian distribution  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbb{1})$ .

#### 6.3.1. Forward process

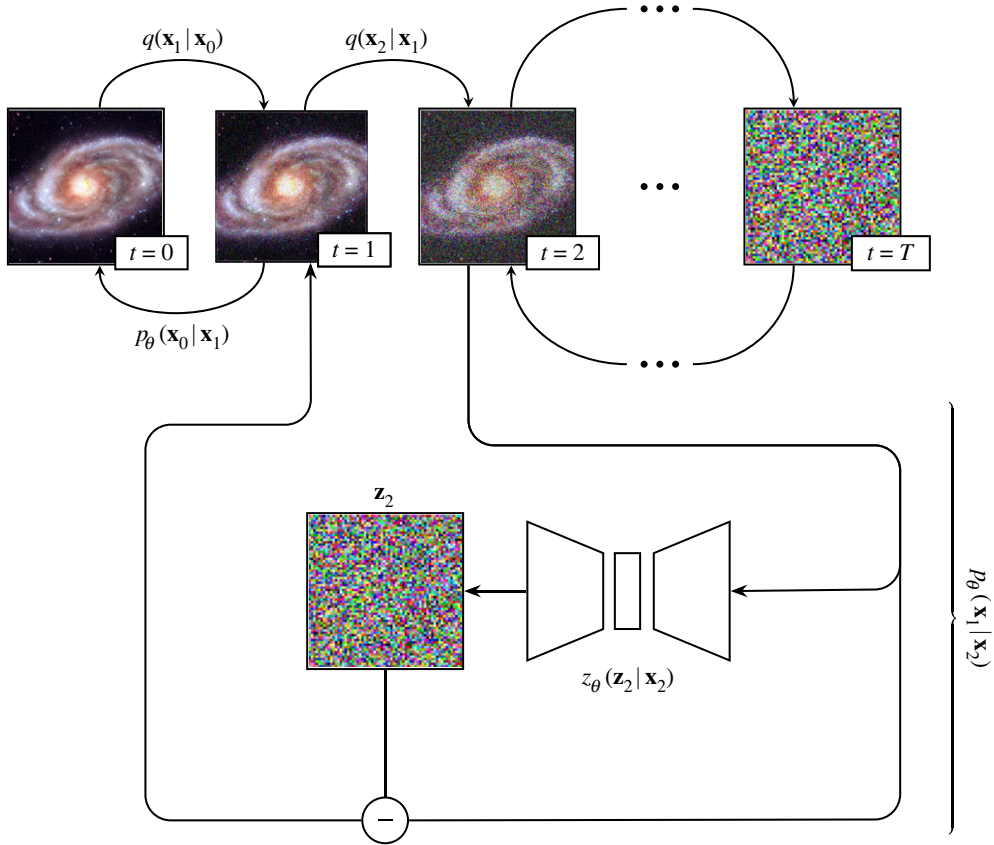
To slowly add Gaussian noise to our data, we define a Markov chain

$$q(\mathbf{x}_{0:T}) = q(\mathbf{x}_0) \prod_{t=1}^T q(\mathbf{x}_t | \mathbf{x}_{t-1}),$$

where  $\mathbf{x}_0$  is an image sampled from the training set. The amount of noise added per step is controlled with a variance schedule  $\{\beta_t \in (0, 1)\}_{t=1}^T$

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbb{1}). \quad (6.3)$$

This process is applied incrementally to the input image. Since we can define the above equation such that it only depends on  $\mathbf{x}_0$  we can immediately calculate an image representation  $\mathbf{x}_t$  for any  $t$  [194]. If



**Figure 20.** It is easy (and achievable without learnt parameters) to add noise to an image, but more difficult to remove it. Diffusion models attempt to learn an iterative removal process via training an appropriate neural network,  $p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$ .

we define  $\alpha_t = 1 - \beta_t$  and  $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$ :

$$\begin{aligned}
 \mathbf{x}_t &= \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \mathbf{z}_{t-1} \\
 &= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{(1 - \alpha_t) + \alpha_t(1 - \alpha_{t-1})} \bar{\mathbf{z}}_{t-2} \\
 &= \sqrt{\alpha_t \alpha_{t-1} \alpha_{t-2}} \mathbf{x}_{t-3} + \sqrt{(1 - \alpha_t \alpha_{t-1}) + \alpha_t \alpha_{t-1} (1 - \alpha_{t-2})} \bar{\mathbf{z}}_{t-3} \\
 &= \dots \\
 &= \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \mathbf{z},
 \end{aligned} \tag{6.4}$$

where  $\mathbf{z}_t \sim \mathcal{N}(\mathbf{0}, \mathbb{1})$  and  $\bar{\mathbf{z}}$  is a combination of Gaussians. Plugging the above expression into equation (6.3) removes the  $\mathbf{x}_{t-1}$  dependency and yields

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbb{1}). \tag{6.5}$$

### 6.3.2. Reverse process

Diffusion models attempt to reverse the forward process by applying a Markov chain with learnt Gaussian transitions. These transitions can be learnt via an appropriate neural network,  $p_\theta$

$$p_\theta(\mathbf{x}_{0..T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$$

and

$$p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)).$$

While  $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)$  can be learnt (e.g. [201]), the Ho *et al.* [194] formulation fixes  $\boldsymbol{\Sigma}_\theta$  to an iteration-dependent constant  $\sigma_t^2 \mathbb{1}$ , where  $\sigma_t^2 = 1 - \alpha_t$ .

By recognizing that diffusion models are a restricted class of hierarchical VAE,<sup>28</sup> we see that we can train  $p_\theta$  by optimizing the evidence lower bound (ELBO, introduced in [175]) that can be written as a summation over the KL divergences at each iteration step<sup>29</sup>

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_q \left[ D_{\text{KL}}(q(\mathbf{x}_T | \mathbf{x}_0) \| p(\mathbf{x}_T)) + \sum_{t>1} D_{\text{KL}}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)) + \log p_\theta(\mathbf{x}_0 | \mathbf{x}_1) \right]. \quad (6.6)$$

In the Ho *et al.* [194] formulation, the first term in equation (6.6) is a constant during training and the final term is modelled as an independent discrete decoder. This leaves the middle summation. Each summand can be written as

$$\mathcal{L}(\boldsymbol{\mu}_t, \boldsymbol{\mu}_\theta) = \frac{1}{2\sigma_t^2} \|\boldsymbol{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, t)\|^2, \quad (6.7)$$

where  $\boldsymbol{\mu}_\theta$  is the neural network's estimation of the forward process posterior mean  $\boldsymbol{\mu}_t$ . In practice, it would be preferable to predict the noise addition in each iteration step ( $\mathbf{z}_t$ ), as  $\mathbf{z}_t$  has a distribution that by definition is centred about zero, with a well-defined variance. To this end, we can define  $\boldsymbol{\mu}_\theta$  as

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \mathbf{z}_\theta(\mathbf{x}_t, t) \right), \quad (6.8)$$

and by combining equations (6.7) and (6.8) we get

$$\begin{aligned} \mathcal{L}(\mathbf{z}_t, \mathbf{z}_\theta) &= \frac{1}{2\sigma_t^2} \left\| \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \mathbf{z}_t \right) - \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \mathbf{z}_\theta(\mathbf{x}_t, t) \right) \right\|^2 \\ &= \frac{(1 - \alpha_t)^2}{2\sigma_t^2 \alpha_t (1 - \bar{\alpha}_t)} \|\mathbf{z}_t - \mathbf{z}_\theta(\mathbf{x}_t, t)\|^2. \end{aligned} \quad (6.9)$$

Ho *et al.* [194] empirically found that a simplified version of the loss described in equation (6.9) results in better sample quality. They use a simplified version of equation (6.9) as their loss, and optimize to predict the noise required to reverse a forward process iteration step:

$$\mathcal{L}(\mathbf{z}_t, \mathbf{z}_\theta) = \|\mathbf{z}_t - \mathbf{z}_\theta(\mathbf{x}_t, t)\|^2, \quad \text{where } \mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \mathbf{z}_t. \quad (6.10)$$

By recognizing that  $\mathbf{z}_t = \sigma_t^2 \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t | \mathbf{x}_{t-1})$ , we see that equation (6.10) is equivalent to denoising score matching over  $t$  noise levels [196]. This connection establishes a link between diffusion models and other SBGMs (such as [197,198,210]).

To run inference for the reverse process, one progressively removes the predicted noise  $\mathbf{z}_\theta$  from an image. The predicted noise is weighted according to a variance schedule

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \mathbf{z}_\theta(\mathbf{x}_t, t) \right) + \boldsymbol{\sigma}_t \mathbf{z}.$$

If we take  $p(\mathbf{x}_T) \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbb{1})$ , we can use  $p_\theta$  to generate entirely novel data that are similar—but not identical to—those found in the training set.

In practice, diffusion models are trained by sampling an integer value of  $t \sim \mathcal{U}(1, T)$ , where  $T$  is a large value typically in the thousands. We then use equation (6.5) to sample an image  $\mathbf{x}_t$  that has had noise added to it  $t$  times. The model then attempts to predict the exact noise required to reverse a forward iteration time step—that is, the output of a neural network<sup>30</sup> of the form  $\mathbf{z}_\theta(\mathbf{z}_t | \mathbf{x}_{t-1})$ . As shown in figure 20, we can estimate  $\mathbf{x}_t$  by removing the predicted noise from  $\mathbf{x}_{t-1}$ . To optimize the model,  $\mathbf{z}_t$  is compared via equation (6.10) with the actual noise required to reverse the forward iteration, and this is the loss that is reduced during training. For a detailed astronomical example with code, we direct the reader to Smith *et al.* [13].

<sup>28</sup>Denoising autoencoders (§6.1) have an interesting relationship with score-based generative (or diffusion) models. As a taster, Turner [208] reframe diffusion models as a class of hierarchical denoising VAE, and Dieleman [209] show through a brief derivation that diffusion models optimize the same loss as a denoising autoencoder.

<sup>29</sup>See appendix B in Sohl-Dickstein *et al.* [193] and appendix A in Ho *et al.* [194] for the full derivation.

<sup>30</sup>Typically, a U-Net; see §4.3.3 for more detail.

### 6.3.3. Denoising diffusion implicit models

Ho, Jain and Abbeel's diffusion model performs inference at a rate orders of magnitude slower than single-shot generative models like the VAE (§6.1) or the GAN (§6.2). This is because diffusion models need to sequentially reverse every step in the forward process Markov chain. Reducing the inference time for diffusion models is an active area of research [199,211,212], and here we will review one proposed solution to the problem; the denoising diffusion implicit model (DDIM, [213]).

Song *et al.* ([213], §§3–4) propose the following reparametrization of equation (6.4):

$$\begin{aligned} \mathbf{x}_{t-1} &= \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \mathbf{z}_\theta^{(t)} + \sigma_t \mathbf{z}_t \\ &= \underbrace{\sqrt{\bar{\alpha}_{t-1}} \left( \frac{\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \mathbf{z}_\theta^{(t)}}{\sqrt{\bar{\alpha}_t}} \right)}_{\mathbf{x}_0 \text{ prediction}} + \underbrace{\sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \mathbf{z}_\theta^{(t)}}_{\text{vector towards } \mathbf{x}_t} + \underbrace{\sigma_t \mathbf{z}_t}_{\text{noise}}, \end{aligned}$$

where  $(t)$  is noted as a superscript to denote the output of the neural network  $\mathbf{z}_\theta$  at time step  $t$ . Intuitively, the first term can be thought of as the prediction of the input image  $\mathbf{x}_0$ , given an iteration step  $t$ . The second term can be thought of as a vector from  $\mathbf{x}_{t-1}$  towards the current iteration step image  $\mathbf{x}_t$ . The third term is random noise. If we substitute in  $\mathbf{x}_t$  from equation (6.10), we make this intuition explicit,

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \mathbf{x}_0}{\sqrt{1 - \bar{\alpha}_t}} + \sigma_t \mathbf{z}.$$

If we then set  $\sigma_t = 0$ , we remove the noise dependency and the forward process becomes deterministic,

$$q_{\text{DDIM}}(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) = \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1}} \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t} \mathbf{x}_0}{\sqrt{1 - \bar{\alpha}_t}}. \quad (6.11)$$

This means that DDIMs can deterministically map to and from the latent space, and so inherit all the benefits of this property. For example, two objects sampled from similar latent vectors share high-level properties, latent space arithmetic is possible, and we can perform meaningful interpolation within this space. We demonstrate DDIM latent space interpolation in figure 21.

We can also subsample every  $\tau$  number of steps at inference time, where  $\tau$  is a set of evenly spaced steps between 0 and  $T$ , the maximum number of steps in the forward process,

$$q_{\text{DDIM}}(\mathbf{x}_{\tau-1} | \mathbf{x}_\tau, \mathbf{x}_0) = \sqrt{\bar{\alpha}_{\tau-1}} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{\tau-1}} \frac{\mathbf{x}_\tau - \sqrt{\bar{\alpha}_\tau} \mathbf{x}_0}{\sqrt{1 - \bar{\alpha}_\tau}}. \quad (6.12)$$

As shown in Song *et al.* [213], this results in acceptable generations with a  $T/\tau$  inference speed-up.

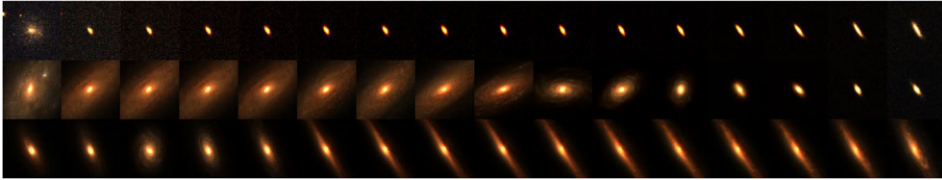
SBGMs have emerged as a promising alternative to GANs, VAEs and other generative models, showcasing their ability to generate high-quality samples with a level of detail comparable to that of the previous state of the art [201–203]. One of the key advantages of SBGMs is how easy they are to train; they do not inherit any of the instability issues that plague GANs. However, SBGMs do have their share of weaknesses. For instance, the SBGM sampling process is computationally expensive and slow. This is because generating a single sample requires a pass through a learnt Markov chain (figure 20), which can limit their practicality in certain applications. Finally, diffusion models and other SBGMs have not been as extensively explored in the deep learning literature as VAEs and GANs (although this is changing fast!). This leaves their applicability across various domains still under investigation.

## 7. Representation learning

Self-supervised<sup>31</sup> representation learning has recently exploded in popularity, with a slew of models being developed in rapid succession (e.g. [214–219]). At its core, representation learning attempts to produce semantically meaningful compressed representations (or embeddings) of complex highly dimensional data. Aside from simply being a compression device, these embeddings can also be taken and used in downstream tasks, like clustering, anomaly detection or classification.

<sup>31</sup>A model that employs self-supervised learning is one that obtains a supervisory signal from the data itself. 'Self-supervised learning' as a descriptor has largely superseded the older term 'unsupervised learning'. This is because the older term suggests that there is no supervisory signal at all—but the signal is there, just not explicitly defined by a human expert!





**Figure 21.** Meaningful latent space interpolation via a DDIM model [13,213]. This property comes ‘for free’ with most other generative models; however, the denoising diffusion probabilistic model [194] requires a tweak to its sampling scheme (equation (6.11)).

In this section, we will describe two approaches to representation learning that are popular within astronomy. The first approach uses contrastive learning as defined by the SimCLR model. The second approach defines and uses a ‘surrogate task’ (such as autoencoding or next-value prediction) to train a deep learning model, and extracts semantically meaningful representations from the subsequent trained network.

## 7.1. Contrastive learning

Figure 22 describes a simple contrastive learning model similar to SimCLR [214]. This model takes as input a sample  $\mathbf{x}$  from the training set, and augments it to produce  $\mathcal{A}(\mathbf{x})$ . This augmentation is performed in such a way that  $\mathcal{A}(\mathbf{x})$  shares enough semantically meaningful data with  $\mathbf{x}$  to belong to the same class. In the contrastive learning literature  $(\mathbf{x}, \mathcal{A}(\mathbf{x}))$  is known as a positive pair. This positive pair is passed to a Siamese neural network  $\Phi$ , which projects the high-dimensional input data onto a lower-dimensional ‘embedding space’. All other training set samples are assumed to belong to a different class to  $\mathbf{x}$ , and so can be combined with  $\mathbf{x}$  to produce ‘negative pairs’. Once we produce some embeddings we need to define a loss that clusters similar samples together, while simultaneously pushing away dissimilar samples. Hadsell *et al.* [220] propose such a loss—the maximum margin contrastive loss

$$\mathcal{L}(\mathbf{z}_i, \mathbf{z}_j) = \delta_{y_i, y_j} d(\mathbf{z}_i, \mathbf{z}_j) + (1 - \delta_{y_i, y_j}) \max(0, m - d(\mathbf{z}_i, \mathbf{z}_j)),$$

where  $\delta$  is the Kronecker delta,  $\mathbf{z}_i$  and  $\mathbf{z}_j$  are embedding vectors,<sup>32</sup>  $y_i$  and  $y_j$  are the class labels for the embedding vectors, and  $m$  is the margin.  $d$  is a ‘distance metric’ (such as for example the L1 loss) that reduces to zero in the case where its inputs are identical. If  $\mathbf{z}_i$  and  $\mathbf{z}_j$  are a positive pair, the loss pulls the embeddings closer, and if they are a negative pair the loss pushes the embeddings away from each other. The margin imposes an upper distance bound on dissimilar embeddings.

While useful, the maximum margin contrastive loss does not take into account the embedding space beyond the pair it is attending to in each training step. This limitation ultimately results in a less expressive embedding space. The triplet loss [221] solves this issue by taking into account the broader embedding space and simultaneously attracting a positive pair while repulsing a negative pair with each training step,

$$\mathcal{L}(\mathbf{z}_i, \mathbf{z}_j, \mathbf{z}_k) = \max(0, d(\mathbf{z}_i, \mathbf{z}_j) - d(\mathbf{z}_i, \mathbf{z}_k) + m), \quad (7.1)$$

where  $\mathbf{z}_k$  is a sampled from a different class to  $\mathbf{z}_i$ , and  $\mathbf{z}_j$  is sampled from the same class as  $\mathbf{z}_i$ .

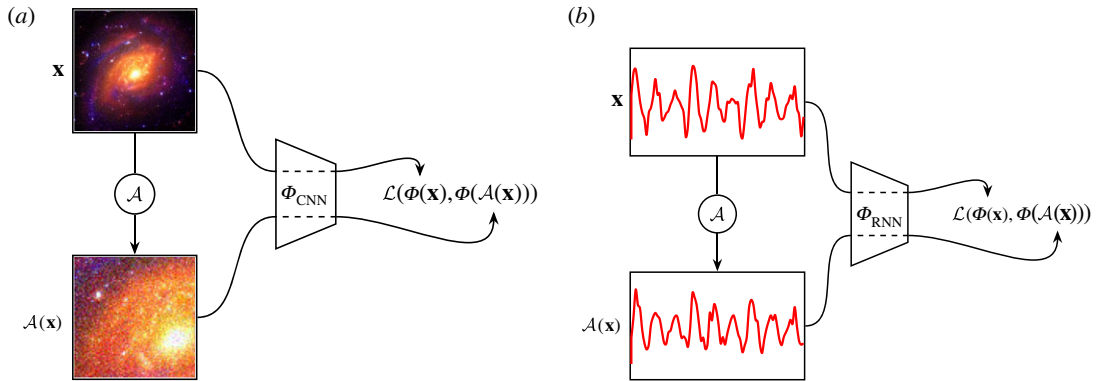
If we study equation (7.1), we see that it is possible to generalize our loss even further, taking into account an arbitrary number of negative samples. The normalized temperature-scaled cross-entropy loss (NT-Xent; [222]) does precisely this,

$$\mathcal{L}(\mathbf{z}_i, \mathbf{z}_j) = -\log \left( \frac{\exp(d(\mathbf{z}_i, \mathbf{z}_j)/\mathcal{T})}{\sum_{k=1}^{2N} (1 - \delta_{ik}) \exp(d(\mathbf{z}_i, \mathbf{z}_k)/\mathcal{T})} \right), \quad (7.2)$$

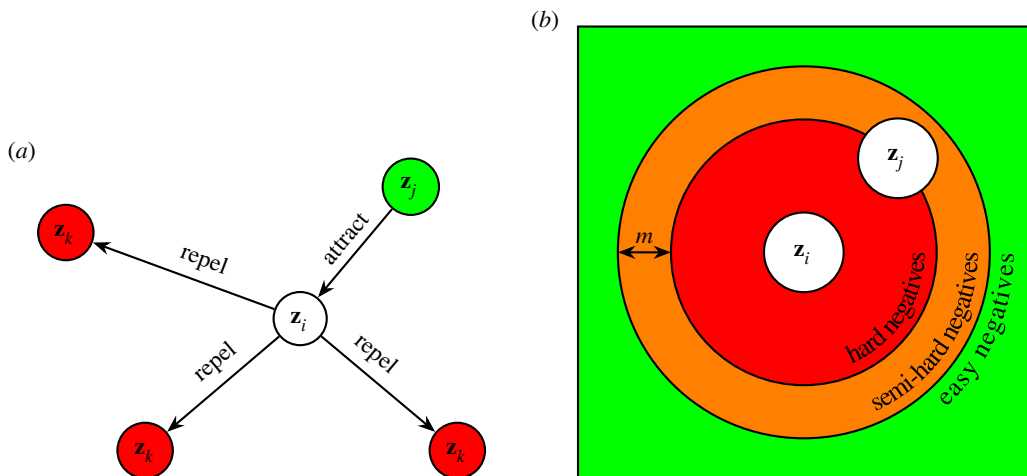
where  $\mathbf{z}_i$  and  $\mathbf{z}_j$  are a positive embedding pair, and  $\mathbf{z}_i$  and  $\mathbf{z}_k$  are a negative pair.  $\mathcal{T}$  is a ‘temperature’ hyperparameter introduced in Chen *et al.* [214] to help the model learn from hard negatives (negatives closer to the anchor than the comparison positive, see figure 23*b*).

<sup>32</sup>All embeddings in this subsection are normalized.





**Figure 22.** A simple contrastive learning model is applied to both imagery and sequential data.  $\mathcal{A}$  is an augmentation pipeline. For imagery,  $\mathcal{A}$  could consist of random crops, noise addition, and colour jitter. For sequential data,  $\mathcal{A}$  could consist of noise addition, stochastic temporal shifting, and random data deletion.  $\Phi$  is a function approximator that projects inputs onto an embedding space.  $\Phi$  is typically a neural network: when processing imagery,  $\Phi$  could take the form of a CNN, and when processing sequential data  $\Phi$  could be an RNN. The loss  $\mathcal{L}$  measures the distance between the embeddings  $\Phi(\mathbf{x}) = \mathbf{z}_i$  and  $\Phi(\mathcal{A}(\mathbf{x})) = \mathbf{z}_j$ , and we train by attempting to minimize this distance while maximizing the distance between dissimilar samples. (a) Possible application to imagery. (b) Possible application to sequential data.

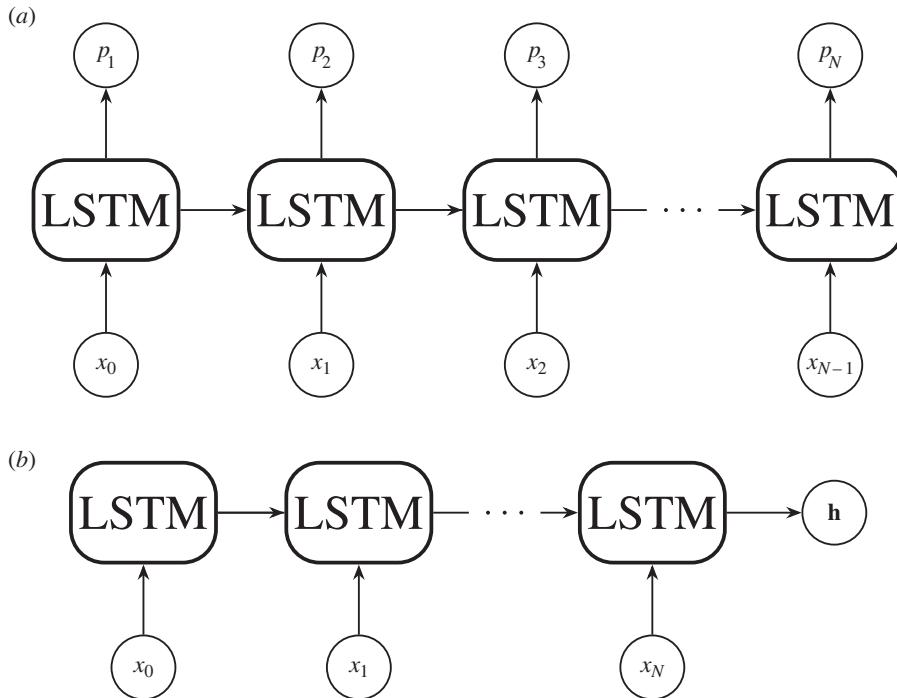


**Figure 23.** (a) The triplet (equation (7.1)) and NT-xent (equation (7.2)) losses simultaneously incentivize attraction between embeddings sampled from the same class ( $\mathbf{z}_i$  and  $\mathbf{z}_j$ ), and repulsion between embeddings sampled from different classes ( $\mathbf{z}_i$  and  $\mathbf{z}_k$ ). (b) Types of negative embeddings.  $\mathbf{z}_i$  and  $\mathbf{z}_j$  form a positive embedding pair. If a negative is closer than the current positive it is considered a hard negative, if it lies within the margin it is considered a semi-hard negative, and if it is beyond the margin it is considered an easy negative.

## 7.2. Learning representations via a surrogate task

One can also learn representations via a surrogate task. A surrogate task is any task that is unrelated to the network's final use. However, in the process of learning to perform the surrogate task, the network learns what is important, and what is unimportant about data within the training set. This information can then be extracted in the form of learnt representations. If the surrogate task is general enough, these representations will contain useful semantic information about the items in the dataset, and can then be used for downstream applications.

Let us concretize this process by revisiting an example that we previously discussed in §4.2. Let us imagine we have a large set of galaxy rotation curves that we want to extract embeddings from. We could train an LSTM model (figure 24) on the task of predicting the next item in the rotation curve, with the model only having access to the previous items in the profile. Once the LSTM model is trained on this task, we can feed in a full, new rotation curve and repurpose the final hidden state as



**Figure 24.** A hypothetical surrogate task for extracting rotation curve representations is shown.  $\{x_0, \dots, x_N\}$  is a set of observations from a galaxy rotation curve, in order of radial distance from the galactic centre.  $\{p_1, \dots, p_N\}$  is the LSTM's corresponding set of predictions for  $\{x_1, \dots, x_N\}$ .  $h$  is the LSTM hidden state vector. See figure 11 for more about the internal workings of the LSTM. (a) While training we feed in the galaxy rotation curve, and predict the next observation in its sequence. (b) While inferring we feed in the full galaxy rotation curve, and extract the LSTM hidden state as a compressed representation embedding of the curve. Otherwise, we ignore whatever output (i.e.  $\{p_1, \dots, p_N\}$ ) the LSTM generates.

a representative embedding. Note that this set-up does not rely on any external labels, only on the rotation curve itself.<sup>33</sup>

We can generate embeddings via an autoencoding task. Again, let us use an astronomical example to specify this and say that we want to extract embeddings from a set of galaxy observations. We could repurpose a variational autoencoder for this, training it as normal as described in §6.1. However, once the model is trained we would discard the decoding part of the network and only consider the encoder. To generate embeddings, we would then simply pass in our galaxy images to the trained encoder. The same process can be carried out by a GAN (§6.2). In the GAN case, we would discard the generator after training and use the discriminator's penultimate layer outputs as our embeddings.

Supervised networks can also be used to generate embeddings. If a network has been trained in a supervised manner to classify or regress data, it will have learnt some properties about that data that helps it to carry out its task. We can access these learnt representations by taking the outputs from a trained network's penultimate layer as an embedding.<sup>34</sup>

## 8. Astronomy's third wave of connectionism

Since its astronomical debut in the mid-2010s [176],<sup>35</sup> deep generative modelling has become a popular subfield within astronomical connectionism. This popularity is driven by its inherent scalability; the lack of a need for labelled data allows the methods to be repurposed for any dataset that might be at hand. Self-supervised connectionism has been around for longer (i.e. [227]), but again has recently exploded in

<sup>33</sup>This self-supervised training set-up is similar to that used to train autoregressive foundation models. These models will be explored in detail in §9.1.

<sup>34</sup>Interestingly, this process is used in the calculation for the Fréchet inception distance (FID) [223,224]. The FID acts as a measurement of the visual similarity between two datasets. The FID works by taking the penultimate layer representations from a trained Inception v. 3 model [225] for each dataset and calculating the distance between them.

<sup>35</sup>Also compare its companion paper [226].

popularity due to its usefulness in wrangling enormous unlabelled datasets. This section is split into two major parts. We will first outline the history of deep astronomical generative modelling in §8.1, and the history of astronomical representation learning will be discussed in §8.2. Although representation learning is the explicit goal for only the studies described in §8.2, it must be stressed that *representations can also be extracted from all the deep generative models described in §8.1.*

## 8.1. Deep astronomical generative modelling

Capturing genuine astronomical data demands accurate knowledge of telescope behaviour, equipment features, environmental factors during observations and data reduction techniques. These complex steps are often tailored to individual observation sets. However, there is an alternative to classical simulation: leveraging examples from a specific survey allows for the development of a data-driven method to simulate not only the astronomical signal but also the inherent characteristics of data. In addition to this, deep learning models trained to replicate astronomical observations are much cheaper to run than classical simulation and so can rapidly generate massive amounts of data; data that can then be used for astronomical pipeline prototyping at scale, aiding the development of new analysis methods, and for dataset augmentation. Data-driven simulation is made possible via the power of deep generative models, and this section describes the history of their use within astronomy.

Seminally, Regier *et al.* [176] proposed the use of a VAE to model galaxy observations. They trained their network on downsampled  $69 \times 69$  crops of galaxies from a SDSS-sampled dataset containing 43 444 galaxies. They trained their network in the same way as described in §6.1, and find that the network is capable of generating galaxies similar to those found in the training set. They also find that their network produces semantically meaningful embeddings, noting that their galaxies are clustered by orientation and morphological type. This same line of enquiry was followed by Ravanbakhsh *et al.* [228], who showed that VAEs could be used to generate galaxies conditionally. Ravanbakhsh *et al.* [228] also pioneered the use of GANs to generate galaxy imagery. Spindler *et al.* [177] used a VAE combined with a Gaussian mixture model prior (see equation (6.2) and accompanying text) to generate and cluster galaxy images into morphological types. While the previous studies in this paragraph used images with relatively small pixel dimensions in their training set, Fussell & Moews [229] and Holzschuh *et al.* [230] demonstrated that GANs are capable of generating large high-fidelity galaxy observations. Fussell & Moews [229] achieved this with a stacked GAN architecture [231], and Holzschuh *et al.* [230] use the related StyleGAN architecture [189] to the same end. Bretonnière *et al.* [12] use a flow-based model<sup>36</sup> [233,234] to conditionally simulate galaxy observations. They found that their approach could produce more accurate simulations than the previous analytical approach, at the cost of inference time. Relatedly, Smith *et al.* [13] use a diffusion model to generate large high-fidelity galaxies. They trained their network on two datasets comprising galaxies as observed by the Dark Energy Spectroscopic Instrument (DESI, [235]). One, a set of 306 006 galaxies catalogued in the SDSS Data Release 7 [81,236,237], and the other a set of 1962 late-type galaxies, as catalogued in the Photometry and Rotation curve OBservations from Extragalactic Surveys (PROBES, [238]) dataset. PROBES contains well-resolved galaxies that exhibit spiral arms, bars and other features characteristic of late-type galaxies. They found that their model produces galaxies that are both qualitatively and statistically indistinguishable from those in the training set, proving that diffusion models are a competitive alternative to the more established GAN and VAE models for astronomical simulation. From all of these studies, we can conclude that deep generative models can internalize a model capable of physically and morphologically describing galaxies.

Generative models have also been used to simulate astronomical data on larger scales. In a use case tangential to galaxy generation, Smith & Geach [239] show that a Spatial-GAN [240] can simulate arbitrarily wide field surveys. They train on the Hubble eXtreme Deep Field, and find that galaxies ‘detected’ within their model’s synthetic deep fields are statistically indistinguishable from the real thing. Cosmological simulations have also been explored, with Rodriguez *et al.* [241] using a GAN to generate cosmic web simulations at pace, and Mustafa *et al.* [242] generating weak lensing convergence maps at a pace faster than classic simulations. Beyond GANs, Remy *et al.* [243]<sup>37</sup> trained a SBGM on simulated maps from MassiveNus [245], and found that their model was capable of replicating these maps. They also demonstrated that their model was capable of producing a likely

<sup>36</sup>Flow-based models have not been discussed in detail in this review, but see Weng [232] for a magisterial introduction to the subject.

<sup>37</sup>This preliminary work has been subsequently extended in Remy *et al.* [244].

spread in the posterior predictions. Finally, they demonstrate that a SBGM is capable of predicting the mass map of the real Hubble Cosmic Evolution Survey (COSMOS) field [246].

The image domain translation abilities of GANs in a Pix2Pix-like formulation ([184], also see figure 19b) is particularly useful in astronomy. Schawinski *et al.* [247] demonstrated this use first by training a Pix2Pix-like model to denoise astronomical data. They trained their network on 4550 galaxies sampled from SDSS. The galaxies were convolved to increase the seeing, and speckle noise was added. The GAN was tasked with reversing this process. They found that their method outperformed both blind deconvolution, and Lucy–Richardson deconvolution. Generative models are also capable of separating sources, as Stark *et al.* [248] demonstrate by using a Pix2Pix model to deblend a quasar’s point source emission from the extended light of its host galaxy. Reiman & Göhre [249] use a similar model to Stark *et al.* [248] to deblend overlapping galaxies.

At the time of writing, there are only three examples of score-based (or diffusion) modelling in the astronomy literature [13,243,244].<sup>38</sup> It is surprising that these studies are the only examples of score-based modelling in astronomy, as SBGMs produce generations that rival that of state-of-the-art GAN models, without drawbacks present in other models (like blurring in the case of VAEs, or mode collapse and training instability in the case of GANs). SBGMs also have some natural uses in astronomical data pipelines. For example, an implementation similar to Sasaki *et al.* [206] could be used for survey-to-survey photometry translation similarly to Buncher *et al.* [254]. The source image separation model described in Jayaram & Thickstun [207] has the obvious application as an astronomical object deblender (i.e. [248,249,255]). To summarize, SBGMs are ripe for exploitation by the astronomical community, and we hope to see much interest in this area in the coming years.

## 8.2. Self-supervised astronomical representation learning

In 1993, Serra-Ricart *et al.* [227] proposed using an autoencoder to learn embeddings for stars as observed by the Two Micron Galactic Survey [256]. They first proved that their autoencoder model worked better than principal component analysis (PCA) on the toy problem of separating Gaussian distributions, and they then showed that their model also outperformed the classic PCA method on real data. More than 20 years later, Graff *et al.* [257]<sup>39</sup> showed that autoencoders are also capable of capturing the properties of galaxies as described in the Mapping Dark Matter Challenge [258] by demonstrating that embeddings extracted from their autoencoder were beneficial for computing the ellipticities of their galaxies as a downstream task. We are not limited to imagery; Yang & Li [259] show that an autoencoder can learn representations that can then be used to train a neural network for the downstream task of estimating stars’ atmospheric parameters, and Tsang & Schultz [260] demonstrate that an autoencoder can generate embeddings that can then be used to classify variable star light curves. From these studies we must conclude that neural networks trained via a surrogate task are capable of learning semantically meaningful embeddings across astronomical domains.

Very recently, there has been work applying self-supervised contrastive learning models to galaxy image clustering. Hayat *et al.* [11] trained SimCLR [214] on multi-band galaxy photometry from the SDSS [81]. They show that the resulting embeddings capture useful information by directly using them in a training set for a galaxy morphology classification model, and a redshift estimation model. Similarly, Sarmiento *et al.* [261] trained SimCLR on integral field spectroscopy data captured from galaxies in the Mapping Nearby Galaxies at Apache Point Observatory survey (MaNGA, [262]). Again, they find that SimCLR produces semantically meaningful embeddings. Slijepcevic *et al.* [263] demonstrate that the ‘Bootstrap Your Own Latent’ (BYOL, [216])<sup>40</sup> contrastive learning model is capable of learning semantically meaningful representations of radio galaxies. Their model is trained on 100 000 Radio Galaxy Zoo galaxies, and inference is run on the 1256 galaxy strong Mirabest dataset [264]. They find that embeddings derived from their model are semantically meaningful,

<sup>38</sup>Since the first posting of this review there have been several workshop papers presented at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022) on the application of SBGMs to astronomical problems (e.g. [250–252]). Here we will highlight a particularly neat example of diffusion model application: Karchev *et al.* [251] tackle the inverse problem of strong-lensing source reconstruction and prove that a denoising diffusion restoration model (DDRM, [253]) inference scheme alongside an off-the-shelf ‘AstroDDPM’ model [13] can restore galaxies that have been through a lensing process. Remarkably, they achieved this *without any retraining or fine-tuning of the original AstroDDPM model*, demonstrating that generalist pretrained score-based models like that described in Smith *et al.* [13] can easily be repurposed for seemingly out-of-distribution downstream tasks. We will revisit the idea of pretrained models that can be repurposed for downstream tasks when we discuss ‘foundation’ models in §9.

<sup>39</sup>See footnote 45 for commentary of this study in the context of astronomical foundation models.

<sup>40</sup>A contrastive learning framework that unlike SimCLR does not use negative samples to learn an embedding space.

suggesting that self-supervised methods are transferable between disparate surveys. These studies show that contrastive learning is applicable to imagery; further study will be required to demonstrate its effectiveness with other types of astronomical data, such as time-series and volumetric data.

## 9. Foundation models: a fourth astroconnectionist wave?

This review has shown thus far that deep learning has found wide use in astronomy, a use predicated on the availability of enormous amounts of computational power and data. This section looks to the future and predicts an outcome if astronomy continues to follow in the footsteps of other applied deep learning fields. In short, we predict and argue that astronomical connectionism will probably see the removal of expertly crafted deep learning models, to be replaced with an all encompassing ‘foundation’ model. In §9.1, we explore what foundation models are, and their context within deep learning. Section 9.2 then contextualizes these models within astronomy, and suggests actions we can take as a community to realize an astronomical foundation model. Finally, §9.3 demonstrates as a thought experiment a state-of-the-art use case for an astronomical foundation model and explores other theoretical and practical uses and implications within (and beyond) astronomy.

### 9.1. Foundation models

Since its inception, connectionism has followed a path of greater compute and greater generality [91,92]. In that time, human-crafted biases have fallen by the wayside, to be replaced with models and techniques that learn directly from data. Sutton [91] exemplifies this process via the field of speech recognition:

In speech recognition, there was an early competition, sponsored by DARPA [Defense Advanced Research Projects Agency], in the 1970s. Entrants included a host of special methods that took advantage of human knowledge—knowledge of words, of phonemes, of the human vocal tract, etc. On the other side were newer methods that were more statistical in nature and did much more computation, based on hidden Markov models (HMMs). Again, the statistical methods won out over the human-knowledge-based methods. This led to a major change in all of natural language processing, gradually over decades, where statistics and computation came to dominate the field. The recent rise of deep learning in speech recognition is the most recent step in this consistent direction. Deep learning methods rely even less on human knowledge, and use even more computation, together with learning on huge training sets, to produce dramatically better speech recognition systems. As in [computer Go and computer chess], researchers always tried to make systems that worked the way the researchers thought their own minds worked—they tried to put that knowledge in their systems—but it proved ultimately counterproductive, and a colossal waste of researcher’s time, when, through Moore’s Law, massive computation became available and a means was found to put it to good use.

We are seeing this principle play out once again through a new paradigm shift in deep learning, where even the underlying neural network architecture does not matter. Previously, neural networks were adapted for a specific domain via inductive biases injected by researchers, such as convolutions for computer vision, and recurrence for language processing. Now we are seeing transformer networks (see §4.4 and [117]) competing<sup>41</sup> in all deep learning domains applied or otherwise: from language processing [17,123]<sup>42</sup> to computer vision [18,168] to graph learning [267] to protein folding [16] to astronomy [169,170,172]. The transformer’s versatility allows us to take a model trained on one task and apply it to a similar yet different task, a process known as transfer learning. For example, we could train a model on the ‘surrogate’ task of predicting the next word in a sequence, and then apply that model to a similar yet different task of predicting the answer to a geography question. In this example, the first model is known as a ‘foundation’ model, and the downstream model is derived from it. This set-up brings with it some useful advantages. For example, if the foundation model is improved, all downstream tasks also see improvement. Therefore, the need for only one model allows researchers to pool their efforts in a way not possible when resources are split between many projects.

To train a foundation model, we first need to define a surrogate task. As labelled datasets are expensive, and raw data are relatively cheap, the easiest and most scalable way to do this is via

<sup>41</sup>For now! It may be that network architecture does not matter all that much at scale, and that any sufficiently large neural network is adequate. If this is true, we will see the simplest (and most scalable) architectures win out. Although this theory has not yet been rigorously tested, we are currently seeing rumblings that suggest that this is the case (e.g. the section ‘Transformers are not special’ in [265]). Bo [266] stands as a particularly notable example of this hypothesis, showing that an attention-free RNN is capable of matching the performance of a similarly scaled transformer network. Also see footnote 12 for commentary on the performance capabilities of MLPs and transformers.

<sup>42</sup>These models are collectively known in the literature as large language models, or LLMs.



self-supervised learning.<sup>43</sup> Self-supervised learning does not require a human to provide a labelled dataset for training. Instead, the supervisory signal is generated automatically from the raw data. For example, in the context of astronomy this task could be predicting a masked value in a variable star's light curve [169]. Another task could be using an autoencoder (§6.1) to replicate a galaxy observation [177]. A further task could be training within a self-supervised framework, like contrastive learning (§7.1). The important thing about self-supervised learning is that it does not require annotated data. This means that we can leverage vast reserves of raw data (such as textbooks, scraped Internet text, raw imagery, etc.).

Very large models trained on vast amounts of data demonstrate surprising emergent behaviour. For instance, GPT-3 [17] is a 175 billion (B) parameter model that can be 'prompted' to perform a novel task (see figure 25 for more on prompting foundation models). This ability was not shown at all in GPT-3's older, smaller 1.5B parameter sibling [122]. Furthermore, a meta-study described in Wei *et al.* [269] found that larger models suddenly 'unlock' abilities such as arithmetic, translation and understanding of figures of speech once they reach a certain scale. These findings suggest that architectural changes are not required beyond scaling to perform many tasks in natural language processing [92,270]. In figure 25, we see some results from Alayrac *et al.* [268], a model comprising an LLM, and an image encoder. In this figure, we can see that the model is capable of arithmetic, reading, counting and has a broad knowledge (albeit not 'understanding') of art, geography and zoology,<sup>44</sup> and literature. This model comprises a ResNet variant [119,272] to encode imagery, and the Chinchilla LLM [273] to encode and generate text. Chinchilla (and therefore Flamingo) was trained with the surrogate task of predicting the next word in a text sequence, and so none of the emergent properties stated above were explicitly optimized for.

In the next subsection, we will state and explain the need for an astronomical foundation model,<sup>45</sup> not only for astronomy's sake, but also for the sake of openness in deep learning research.

## 9.2. Scaling laws and data moats

Hoffmann *et al.* [273] suggested an update to the foundation model scaling law first proposed in Kaplan *et al.* [275]. Their scaling law equation relates the size of a neural network model and the training dataset size to the minimum achievable loss. Mathematically, the equation is

$$\mathcal{L}_{\min}(N, D) = \underbrace{\frac{A}{N^\alpha}}_{\text{parameter term}} + \underbrace{\frac{B}{D^\beta}}_{\text{data term}} + \underbrace{E}_{\text{dataset entropy}}, \quad (9.1)$$

where  $E$  is a constant that represents the lowest possible loss, given a particular training dataset.  $N$  is the number of trainable parameters within the neural network, and  $D$  is the size of the dataset in tokens (see §4.4 for more about tokenization). We can see that when we have an infinitely large model trained on an infinitely large dataset (i.e.  $N = D = \infty$ ), the only term remaining is the 'dataset entropy' constant,  $E$ . We can therefore only reduce the loss by increasing the size of our model, or the size of our training set.

After fitting equation (9.1), Hoffmann *et al.* [273] find

$$\mathcal{L}_{\min}(N, D) = \frac{406.4}{N^{0.34}} + \frac{410.7}{D^{0.28}} + 1.69.$$


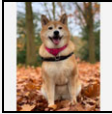


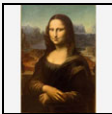

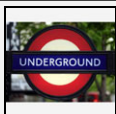


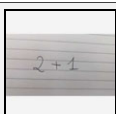
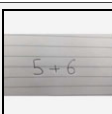
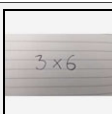



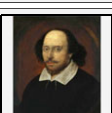
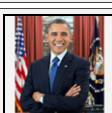
If we then plug in  $N$  and  $D$  for a selection of real foundation models we arrive at figure 26. We can see in figure 26 that the model size term for real foundation models is far lower than the dataset size term. This

<sup>43</sup>For more on self-supervised learning, see §7.

<sup>44</sup>Interestingly, the authors of Flamingo first assumed that Flamingo's prediction of the species range of its eponymous bird was incorrect: flamingos are found in the Caribbean, South America, Africa, Europe and South Asia. However, they later realized that the picture in figure 25 is of an *American* flamingo, which is specifically found in the Caribbean and South America, so the network was right after all! See the reddit thread for the full context [271].

<sup>45</sup>Walmsley *et al.* [274] explore in a preliminary study a 'galaxy foundation model' trained on Galaxy Zoo labels, and corresponding paired galaxy observations. They find that their pre-training is beneficial for training a network that performs a downstream task. However, the idea has been around for far longer than that; possibly the first demonstration of an astronomical foundation model was described 8 years earlier in Graff *et al.* [257]. Graff *et al.* [257] demonstrated that embeddings learnt with their autoencoding SkyNet network can be used for downstream tasks, but they do not use the moniker 'foundation model' to describe SkyNet, as the term had not yet been invented! Notably, neither study trains a model of the scale required to exhibit emergent properties or task generalizability. These 'blessings of scale' require data and compute at a level that has not yet been seen within astronomical connectionism.



input prompt				completion
	This is a chinchilla. They are mainly found in Chile.		This is a shiba. They are very popular in Japan.	→ a flamingo. They are found in the Caribbean and South America.
	This is			
	What is the title of this painting? Answer: The Hallucinogenic Toreador.		Where is this painting displayed? Answer: Louvres Museum, Paris.	→ Arles.
	What is the name of the city where this was painted? Answer:			
	Output: 'Underground'		Output: 'Congress'	→ 'Soulomes'
	Output:			
	$2 + 1 = 3$		$5 + 6 = 11$	→ $3 \times 6 = 18$
				
	pandas: 3		dogs: 2	→ giraffes: 4
				
I like reading		, my favourite play is Hamlet. I also like		→ dreams from my father.
			, my favourite book is	

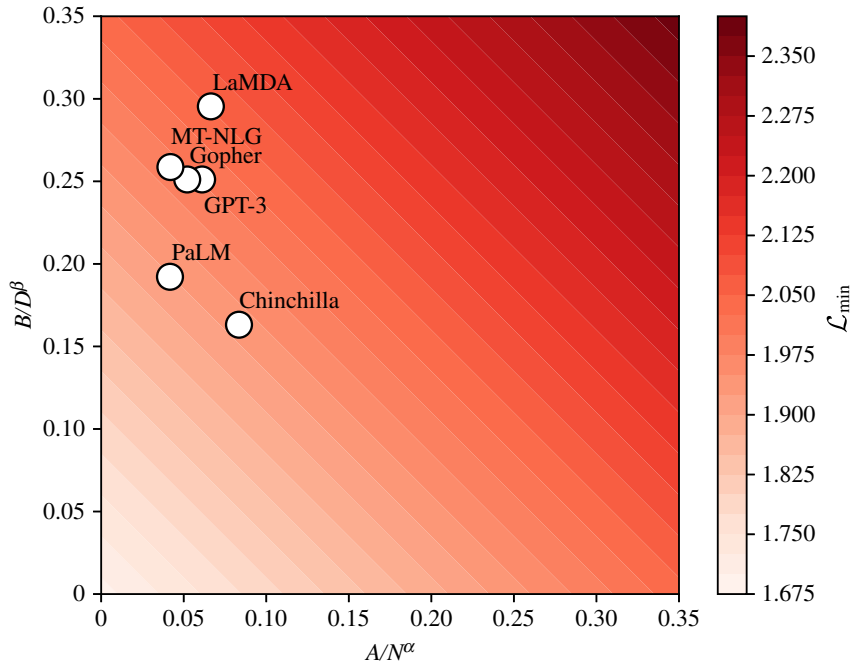
**Figure 25.** Flamingo is a foundation model that is capable of understanding images within the context of natural language. Here we see some examples of Flamingo's emergent abilities. This figure is adapted from fig. 1 in Alayrac *et al.* [268].

means that an increase in dataset size has the potential to reduce the minimum loss by a far larger amount than a larger model would. Therefore, an obvious next step to improve these foundation models further is by increasing their dataset size.

The largest dataset (MassiveText-English; [273]) in the comparison shown in figure 26 amounts to 1.4 trillion (T) tokens. However, this dataset is proprietary, being only available to researchers employed by Google. The largest public text dataset available at the time of writing is The Pile [279], with a total size of approximately 260B tokens. We could increase the size of these datasets by indefinitely scraping text data from the surface web, but these data tend to be of low quality. Also, we have already exhausted some important high-quality data reserves, like fundamental research papers, and open source code [280]. We also have to ask ourselves: what happens when generative models start to create data en masse, and dump it indiscriminately onto the Internet? If a significant proportion of text in a dataset scraped from the Internet is generated via an LLM, training on it will cause unforeseen issues and may ultimately result in a model with worse performance. We must therefore ensure that the data are not generated by a deep generative model. In addition to all this, the academy and the public at large will never have access to the vast reserves of data contained in the deep web administered by ByteDance, Google, Meta, Microsoft and other tech giants. For all these reasons, we will need to think outside the box if we want to mine new high-quality data.

Enter the multi-modal foundation model. Reed *et al.* [124]<sup>46</sup> demonstrated that a large transformer neural network is capable of learning many tasks, from playing Atari, to captioning images, to chatting, to operating a real robot arm. The model shares weights across all tasks, and decides at inference time from context which task to predict. Importantly, Reed *et al.* [124] find that their model follows the same scaling laws as other foundation models, and so multi-modal foundation models have the same hunger for data that we see in figure 26. Even more astonishingly, Aghajanyan *et al.* [282] find that a foundation model trained on concatenated independent datasets significantly

<sup>46</sup>Earlier work from Kaiser *et al.* [281] also demonstrated a deep learning model that could learn from disparate tasks; however, Gato is the first model that achieves this while staying within a single deep learning paradigm.



Model	$N$	$D$	$A/N^\alpha$	$B/D^\beta$	$\mathcal{L}_{\min}$
LaMDA [275]	137B	168B	0.066	0.295	2.051
GPT-3 [17]	175B	300B	0.061	0.251	2.002
Gopher [276]	280B	300B	0.052	0.251	1.993
MT-NLG [277]	530B	270B	0.041	0.259	1.990
Chinchilla [272]	70B	1.4T	0.083	0.163	1.936
PaLM [269]	540B	780B	0.042	0.192	1.924

**Figure 26.** A comparison between the minimum losses of a selection of foundation models. The table above shows the number of parameters in a model ( $N$ ), the number of tokens within that model’s training set ( $D$ ), and their corresponding calculated emergent terms from equation (9.1). Here we use Hoffmann *et al.* [273] to source values for  $A$ ,  $\alpha$ ,  $B$  and  $\beta$ . The minimum loss for each model according to Hoffmann *et al.* [273] is shown as  $\mathcal{L}_{\min}$ . The contour plot shows the emergent parameters  $B/D^\beta$  and  $A/N^\alpha$  plotted against each other for our models. The closer the models’ scatterpoints are to the bottom left, the lower their minimum loss value.

outperforms separately trained unimodal models once the neural networks reach a certain scale. We can therefore augment our text datasets with high-quality, publicly available astronomical data.

The Vera Rubin Observatory’s 189 16 megapixel CCDs will observe 1000 science frames per night while conducting the Legacy Survey of Space and Time (LSST) [283]. This amounts to  $3 \times 10^{12}$  pixels per night, or approximately 12B tokens a night if we use the same tokenizing scheme as Dosovitskiy *et al.*’s vision transformer [18]. After only 1 year of observing, the LSST will have produced 4.4T tokens of raw data, larger than even the MassiveText-English dataset.<sup>47</sup> These data, and other astronomical data like it, could be compiled into a very large open dataset similar to EleutherAI’s Pile [279]. This dataset would provide a way for academics employed outside of Big Tech to train and research very large foundation models. Compiling a dataset like this would be difficult for a single relatively under-resourced research group, but it could be accomplished via bazaar style open development [284]. We have already seen this development model succeed in large open source projects, the most famous of which is the Linux kernel. This development model has also been shown to work within the field of deep learning by EleutherAI (e.g. [279,285,286]), and with HuggingFace’s BigScience initiative [287]. Once compiled, we must ensure that progress is kept in the open, and that the data are not simply absorbed into proprietary datasets—to do this we must give our dataset a strong (viral) copyleft style licence.

<sup>47</sup>Of course, the reduced, useful data will be far smaller than our raw estimate here. The motivation behind this calculation is to show that even a single astronomical survey rivals the largest text dataset in size. A compilation of all useful astronomical data would certainly dwarf any contemporary text dataset, whether public or proprietary.

Once the dataset is compiled all we need for training are some self-supervised surrogate tasks for our ‘astrofoundation’ model to attempt. These tasks could include predicting the next observation in a variable star’s time sequence, predicting the low surface brightness profile of a galaxy, predicting a galaxy’s morphological parameters or simply generating the next crop in a sequence of observations.<sup>48</sup> As we will explore in the next subsection, these surrogate tasks do not need to be at all related to the downstream tasks we will eventually use our model for. Once trained, our astrofoundation model will inherit all the interesting properties that LLMs enjoy, such as few- to zero-shot generation and other emergent behaviours.

### 9.3. The practical implications and uses of an astrofoundation model

This section explores the wider implications of a hypothetical astrofoundation model (§9.3.1), as well as some practical astronomical uses (§9.3.2). In §9.3.3, we highlight one possible downstream task that would be useful in astronomy; a conditional generative model for astronomical simulation.

#### 9.3.1. Democratizing foundation models

The spring of 2023<sup>49</sup> has brought with it a shift in the global zeitgeist’s attention towards foundation models in general, and the GPT family of large language models in particular. Leading the charge is OpenAI’s ChatGPT, whose release has become a very public advertisement of the abilities that large language models possess (figure 27). While impactful, we note that ChatGPT is ‘just’ a web interface wrapper for versions of GPT-3 and GPT-4 that have been fine-tuned using human feedback [291,292]. ChatGPT’s popularity therefore suggests that there is a lot of latent general interest in deep learning and foundation models, and that this interest can be realized through a convincing public demonstration. Fully open development and dissemination of these models is perhaps the most public demonstration there is. And we have indeed seen that the release of open source foundation models leads to an explosion of innovation and interest.<sup>50</sup> One particular example is the release and impact of the ‘large language model [from] Meta AI’ (LLaMA; [293]). The LLaMAs are a collection of open source LLMs, and the largest LLaMA has a comparable performance to GPT-3. Since LLaMA’s release, an entire ecosystem of projects have spun up that use the model in innovative and interesting ways (e.g. [294–297]). A similar story occurred in 2022 when StabilityAI released an open text-to-image diffusion model based on latent diffusion [94]. The following flurry of activity far outstripped the progress OpenAI made with their competing closed source DALL-E 2 model [203,298]. We believe that a similar explosion of innovation to that seen with the release of the LLaMA and Stable Diffusion models would lay in store for astronomy if an open astronomical foundation model is developed and marketed effectively.

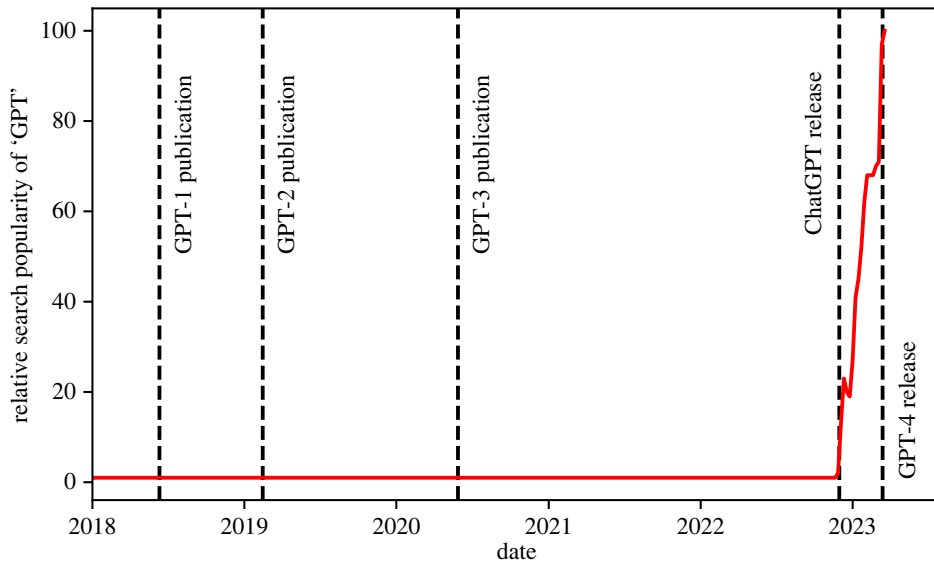
In mid-March GPT-4 was released [26]. Its accompanying ‘Technical Report’ contains no detail on the model’s architecture, training set size, or training routine.<sup>51</sup> The unashamed release of a closed model is quite a worrying development for a field that has historically been built on open source and open research. Of most concern is industrial actors within this space closing up shop as a reaction to the open/closed model prisoner’s dilemma set by OpenAI. As figure 28 shows, industry has produced the lion’s share of impactful deep learning models since the mid-2010s; if future developments are kept hidden due to commercial pressure we will see a concentration of talent and innovation locked away behind industry’s closed doors. Furthermore, the latest developments in foundation modelling have the potential to significantly impact the global economy and workforce through pervasive automation [173,300]. As automation increases, the concentration of power, expertise and economic clout within large industrial actors will weaken the economic bargaining position of those that do not have access to these technologies. This could result in a societal equilibrium where fewer and fewer

<sup>48</sup>This is essentially training the model to act as a physics simulator. Viewing foundation models as world simulators is not unprecedented. This perspective has already been explored in the simulation of thousands of ‘social simulacra’ within a model online community [288], and with the simulation of participants in classic (i.e. Milgram’s shock experiment, the Ultimatum Game) and novel psychological studies [289].

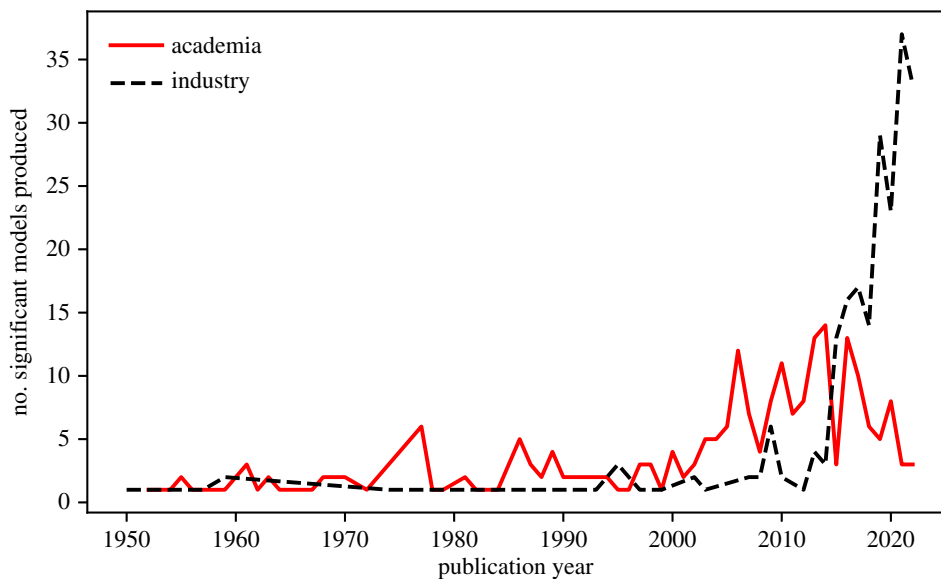
<sup>49</sup>While we revisited this subsection for our review rebuttal.

<sup>50</sup>This is a specific example of the more general rule that ‘bazaar’ (public from conception) style open development outcompetes the ‘cathedral’ model (closed until release, or in this case closed even after release) on an equal playing field [284].

<sup>51</sup>Although if we extrapolate from the historical trend of LLM development, OpenAI’s general research culture and direction and the time GPT-4 takes to run inference, we could arrive at the conclusion that GPT-4 is essentially a scaled-up ‘GPT-3’ model that follows a Chinchilla-optimal scaling law (§9.2).



**Figure 27.** Here we show the relative Google search popularity for the term ‘GPT’. We can see a huge increase in searches for GPT when the ChatGPT model was launched for public use (and surprisingly little increase in searches when the GPT-1, GPT-2 and GPT-3 papers were released!) [17,26,122,290]. These data are taken from Google Trends (<https://trends.google.co.uk/>).



**Figure 28.** Here we show the number of highly cited, state-of-the-art or historically significant works produced per year within academia and industry. These data are taken from Sevilla *et al.* [299].

people have access to economic and social opportunity. This is an equilibrium that Brynjolfsson [301] memetically dubs the ‘Turing Trap’:

A fully automated economy could, in principle, be structured to redistribute the benefits from production widely, even to those who are no longer strictly necessary for value creation. However, the beneficiaries would be in a weak bargaining position to prevent a change in the distribution that left them with little or nothing. They would depend precariously on the decisions of those in control of the technology. This opens the door to increased concentration of wealth and power.

To avoid this trap, we must collectively work towards making foundation models—and by proxy the latest fruits of automation—available to all. A copyleft foundation model trained on a copyleft dataset (such as our hypothetical astronomical foundation model) would go some way towards reducing the growing technological inequality between Big Tech and wider society.

With the above discussion in mind, we would like to revisit our brief analysis in §9.2 and restate and emphasize the pressing need for an independent, verifiable, completely open and strong copyleft licensed

alternative to the closed foundation models controlled by OpenAI, Microsoft, Anthropic, Google and other Big Tech conglomerates. While expensive, the compute is fairly easy to source—the paramount issue is that foundation models require a huge amount of data to train them effectively. These models are usually trained via a large amount of high-quality publicly unavailable textual data that is locked within the deep web. Fortunately, however, §9.2 shows that a large amount of useful multi-modal data can be easily sourced from astronomical observations. We can therefore conclude this subsection on a positive note—astronomy is ideally poised to play an outsized role in the democratization of foundation models.

### 9.3.2. Possible astronomical use cases

In this subsection, we outline some possible exciting astronomical uses for our astrofoundation model. Before we dive in, we must state that here we only skim the surface of this technology’s potential, and we hope that—as evidenced by the LLaMA and Stable Diffusion ecosystems (§9.3.1)—there will be many more use cases that we have not discussed here that would emerge from community involvement. We divide this subsection into two parts. The first part talks about how a foundation model could aid outreach, citizen science and cross-disciplinary collaboration, and the second part discusses how the model could aid astronomical research.

#### 9.3.2.1. Collaboration, citizen science and outreach

By providing a common platform for generating simulations and analysing data, a neural network-based astrofoundation model would ease and facilitate collaboration between researchers in previously disparate fields. In addition to this, any improvement in the underlying technology could easily be integrated into field-specific (or field-agnostic) foundation models that could be used for tasks that previously needed years of specialist training to operate. One example specific to astronomy is astronomical simulation. A physically aware astrofoundation model could be used to simulate and interrogate simulated astronomical events in much the same way that classical simulations do now [20–22]. Section 9.3.3 describes in detail one framework that could facilitate such a model.

The multi-modal training of neural networks lets us make connections between data modes that would be impossible or difficult with current methods. As just one example, let us consider citizen science. In a citizen science project like Galaxy Zoo [132], citizen scientists are asked to label astronomical objects with quantitative labels. This can be an unintuitive process for someone untrained in astronomy. An astronomical foundation model that has an awareness of natural language would allow participants to describe astronomical objects using their own words. This would reduce the need for specialized training and therefore would increase the accessibility of these projects. One could imagine a new Galaxy Zoo-like project where citizen scientists provide natural language descriptions of galaxy morphologies. The foundation model could then process and analyse these descriptions, which would eventually contribute to a more comprehensive understanding of galaxy evolution.<sup>52</sup>

A foundation model with astronomical knowledge could be used to develop chatbots capable of engaging students, educators and the general public in conversations about astronomy. These chatbots could answer questions, provide explanations, or even suggest personalized learning resources based on the user’s interests and prior knowledge. This would widen and democratize access to astronomical knowledge, and this easy access to astronomical knowledge could enthuse and help to recruit the next generation of astronomers. Foundation models can already act as tutors, and commercial actors are currently working in this space; the most notable examples being ‘Duolingo Max’ which provides users a personalized chatbot for foreign language learning, and Khan Academy’s ‘Khanmigo’ which provides students a personal tutor for their courses. Both Duolingo Max and Khanmigo are paid offerings powered by OpenAI’s GPT-4 API [26], and so an open astronomical foundation model would provide wider access than a closed GPT-N model that has been prompted to become astronomically aware.

#### 9.3.2.2. Augmenting research

While the foundation model is necessarily trained on existing data, its ability to identify patterns and relationships within the data can lead to new knowledge discovery, and a more efficient way to process data that previously was difficult or time consuming. As discussed previously in §§6–8, an astroconnectionist could use the foundation model to generate embeddings for a set of astronomical

<sup>52</sup>Work is already being done to realize this. For example, Bowles *et al.* [302] propose a semantic natural language labelling scheme for the Galaxy Zoo evolutionary map of the universe project.

objects. Like we discussed in §§6–8, these embeddings could be used for downstream astronomical tasks, or could be placed into visualization pipelines like the t-distributed stochastic neighbour embedding method [303,304]. Since the astronomical foundation model would be multi-modal, a researcher could combine the embeddings of multiple datasets generated from entirely different instruments, giving them a birds-eye view of their data that would currently be difficult to achieve. We can also use the foundation model’s emergent abilities to our advantage; as shown in figure 25 we could use few-shot learning and prompt the trained model with a few example pairs of inputs. For instance, we could use pairs of input galaxy observations and corresponding output surface brightness profiles [167]. If the astronomical foundation model is a few-shot learner (and is aware of a similar input–output pairing within its training data), it would identify that the researcher wants to calculate the surface brightness profiles of new galaxies. The researcher would then use the prompted model as a surface brightness profile extractor, sidestepping the need for a specialized analytical or deep learning model for such a task. This process is not limited to this example—it would work for any input–output pair within a mode that the foundation model is aware of. Even better, this process would require no retraining of the foundation model, it would only require the few-shot prompt at inference time.

Autonomous agents are no longer science fiction; task-driven autonomous agents powered by the simulacra of a foundation model are capable of solving very general tasks when given only a high-level prompt by a human operator [305,306]. One could therefore imagine a semi-automated research pipeline, where an autonomous agent with astronomical knowledge is given access to a set of astronomical data through an API. The agent would be prompted with a high-level research goal (such as ‘find something interesting and surprising within this dataset’), and would then take steps to achieve this task. These steps could include querying research papers for a literature review, searching a large multi-modal astronomical dataset to find data that supports a theory, evoking and discussing its findings with additional simulacra, or spinning up simulations to test a hypothesis [307]. While the agent operates in the background, the human researcher would be able to provide high-level interpretation of the results, and would be a steady hand providing guidance and refinement of a more general research direction. In this way, an astronomical foundation model would provide the tools to make all astronomers the principal investigator of their own powerful ‘AI lab’.

### 9.3.3. A new class of simulation

We would like to end this subsection with a tangible application of our hypothetical astrofoundation model; a conditional generative model for astronomical simulation in the spirit of recent work on text-to-image modelling (i.e. [94,308]). If we train an unconditional generative model, we cannot control its output at inference time. This is an issue if we want to generate specific classes of observations to train models for downstream tasks, such as redshift estimation, or galaxy-type classification. To achieve a model capable of generating specific classes, one could simply train a conditional generative model of the form

$$G_{\phi}(\hat{\mathbf{x}} \mid \mathbf{z}, \mathbf{y}), \quad (9.2)$$

where  $\hat{\mathbf{x}}$  is a generated image,  $\mathbf{z}$  is some noise that acts to capture all detail not encoded in  $\mathbf{y}$ , and  $\mathbf{y}$  is a conditioning vector. As an example,  $\mathbf{y}$  could contain a galaxy’s redshift or morphological type. However, this means that we must be very specific when choosing  $\mathbf{y}$ . Multi-modal modelling provides us the means to sidestep this fundamental issue, and lets us play with fuzzy inputs.

As a thought experiment, let us consider Google’s recent ‘Imagen’ model,<sup>53</sup> and imagine how it could be repurposed for an astronomical use case (figures 29 and 30, [308]). Imagen is a combination of a frozen LLM (specifically T5-XXL; [310]) and a cascaded diffusion model ([309], also see §6.3). The LLM acts as a language encoder, and then passes its generated latent space representations onto the diffusion model as a conditioning vector. If we were to replace the frozen LLM with an ‘astrofoundation’ model (see §9.1 and 9.2), we could leverage astronomy’s fundamentally multi-modal nature. For example, if our astrofoundation model were trained to understand the Galaxy Zoo 2 (GZ2) morphological classifications [311], we could take the GZ2 descriptors as  $\mathbf{y}$  and their corresponding galaxy pair as  $\mathbf{x}$  and train on those.

<sup>53</sup>Naturally, no implementation is provided by Google. However, there is a fantastic MIT-licensed implementation of Imagen provided by Phil Wang and others (<https://github.com/lucidrains/imagen-pytorch>), and StabilityAI has a similar trained open source model released under the name ‘Stable Diffusion’ (<https://github.com/Stability-AI/stablediffusion>).

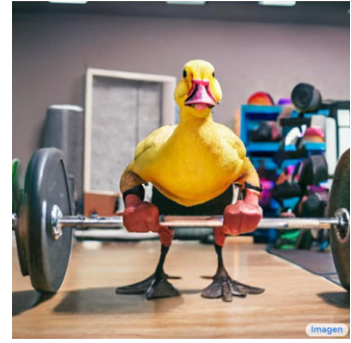




A wall in a royal castle. There are two paintings on the wall. The one on the left a detailed oil painting of the royal raccoon king. The one on the right a detailed oil painting of the royal raccoon queen.



A group of teddy bears in suits in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.



An angry duck doing heavy weightlifting at the gym.



A cloud in the shape of two bunnies playing with a ball. The ball is made of clouds too.

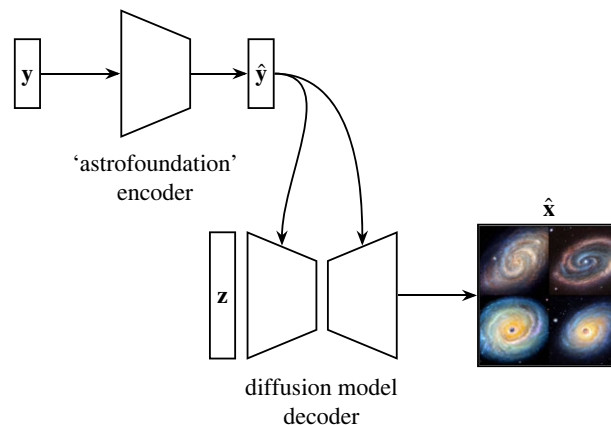


A photo of a person with the head of a cow, wearing a tuxedo and black bowtie. Beach wallpaper in the background.



A chrome-plated duck with a golden beak arguing with an angry turtle in a forest.

**Figure 29.** Select  $1024 \times 1024$  Imagen samples generated from text inputs. Below each image is its corresponding conditioning text. Figure adapted from fig. A.2 in [308].



**Figure 30.** An Imagen-like model uses a frozen foundation model to encode text, and then uses that encoding to condition a cascaded diffusion model of the form  $G_\phi(\hat{\mathbf{x}} | \mathbf{z}, \hat{\mathbf{y}})$  [308,309]. Here we see one possible realization of this type of model in astronomy.  $\mathbf{y}$  is some kind of descriptive vector that can be paired with a ground truth image. For example,  $\mathbf{y}$  could be the surface brightness profile of a galaxy, or the summary statistics of a variable star light curve, or some cosmological parameters. In general,  $\mathbf{y}$  could be any vector that the astrofoundation model understands.  $\hat{\mathbf{y}}$  is  $\mathbf{y}$ 's projected latent space equivalent. Since we do not need to train the foundation model here, training cost is far lower than for an equivalent end-to-end trained model.

Once trained, our astronomical Imagen model could generate synthetic galaxies that resemble the real galaxy observations that it was trained on. However, unlike an unconditional astronomical simulator, this model would be capable of generating galaxies that specifically resemble a real galaxy that shares the conditioning set of GZ2 parameters!

Unlike the conditional model described by equation (9.2), an astrofoundation-type model allows us to be creative with the conditioning vector. For example, we could run the model in reverse to generate representations that refer to a very specific astronomical object, and then generate many more objects of that ‘class’ with injected features like satellite occlusion, a specific instrument response function, a specific redshift, etc. (see work on ‘textual inversion’ by Gal *et al.* [312]). These simulations would enable researchers to create tailored datasets for various research purposes, such as studying particular galaxy types, morphologies or cosmological phenomena. We could even create a ‘Galaxy Zoo’ type dataset that asks citizen scientists to describe galaxy morphology via natural language (§9.3.2). This is possible since the encoding foundation model does not fundamentally care about which form the caption takes. This approach would cut down on citizen scientist training cost due to natural language’s inherent intuitiveness. Furthermore, as inference-time generation is relatively cheap, a model like the one described in this section would allow astronomers to explore and test hypotheses and scenarios more rapidly than they could if they used a classical simulation.

## 10. Connectionism’s caveats

Thus far in this review we have been very optimistic about astronomical connectionism’s potential. However, this does not mean that connectionism is without its pitfalls. Section 10.1 outlines some practical downsides of astronomical connectionism, and discusses how a practitioner can mitigate them. Owing to its importance, we dedicate §10.2 to the discussion of climate change and carbon emissions, and illustrate connectionism’s impact with a case study on the carbon emissions of modern large language and foundation models.

### 10.1. Possible practical pitfalls

As illustrated in figure 26, deep learning has an insatiable hunger for data. Acquiring and labelling data for the training of deep learning models can be extraordinarily expensive and time-consuming. The savvy astroconnectionist could mitigate this problem through self-supervised or generative learning that does not require labelled data, and then repurposing learnt embeddings for more specialized downstream tasks<sup>54</sup> (see §§6–9). Related to this, rare or entirely unexpected astronomical events and phenomena<sup>55</sup> are by definition poorly sampled within any training data, and so a deep learning model will have difficulty generalizing and internalizing these events. One solution is using an anomaly detection method to find these rare phenomena. We direct the reader to Pang *et al.* [315] for an excellent recent review of anomaly detection techniques.

Very large deep learning models can be expensive to train and run inference with. Some astronomical applications, such as detecting transient events, require real-time processing of large volumes of data. The computational complexity of deep learning models can pose challenges for their deployment in these time-sensitive scenarios. In that case, it may be preferable to employ a fast, simple, classical technique or to use a smaller deep learning model.

Astronomical data can be observed via a variety of different instruments (or simulations), and the final output data can be processed by any number of post-processing pipelines. These pipelines each have their own characteristics, idiosyncrasies and foibles, and so can appear very different when propagated through a deep neural network. Also, the distribution of known celestial objects within a survey may be influenced by observational biases or historical interests, and so careful inspection of datasets is required to ensure that they are representative for the desired use case. In addition to care, an astroconnectionist might employ domain adaptation techniques to ensure that their datasets are representative for their downstream tasks [316]. Finally, as we explored in §9, it may even be enough to simply train a very large deep learning model on a collection of datasets [282], but this approach is currently out of reach for the average researcher.

<sup>54</sup>This process is also known as ‘transfer learning’.

<sup>55</sup>Such as Green Bean Galaxies [313], or SETI events akin to the ‘Wow!’ signal [314].

The perennial criticism of deep learning is—of course—interpretability. As deep learning models are highly parametrized it is difficult to understand why they arrive at a certain behaviour or decision. There are many ways to sidestep this issue, and this paragraph will briefly outline some developments in this direction that might be of use to a practitioner. Perhaps the gold standard for interpretability is a neural network walking the user through its ‘thought’ process step-by-step with natural language, as a human would do. Large language foundation models can do this, and this ability comes ‘for free’ with a sufficiently large model and dataset [317]. Unfortunately, however, no such foundation model currently exists that also has a deep knowledge of astronomy (§9) so we must be a little more creative. Attentional mapping can be used to show which features the deep learning model are attending to when producing an output, and this attentional mapping can be depicted as a heat map over our data. Attentional mapping can be generated in several ways; for example, we could use a mechanism like we discussed in §4.4 to highlight the most useful parts of an input datum for the model to predict or generate its output. One can also use class activation mapping [231] to trace the outputs of a fully convolutional neural network back to its inputs to see which parts of an input image are used in a prediction. Occlusion mapping (and other perturbation techniques) can be used to visualize attention for all architectures. Occlusion maps require us to occlude parts of an input datum and in turn allow us observe how that affects the output prediction [137]. We can also apply certain statistical methods to deep learning models to gain an insight into their inner workings. Stochastic neural networks trained within the Bayesian paradigm (or ‘Bayesian neural networks’) can be used to estimate the uncertainty in neural network predictions [318]. One does not need to have prior knowledge of the dataset when training a Bayesian neural network; neural networks can make use of approximate Bayesian computation techniques like likelihood-free inference to estimate the posterior [319]. Besides these methods, many other deep interpretability pipelines are in use—far more than we have space to go over here—and so we highly recommend Ras *et al.* [320] for a general and extensive overview of the field of explainable deep learning.

## 10.2. Connectionism’s carbon crisis

The training of deep learning models in general requires a considerable amount of energy, and it is only natural that the training of ultra-large foundation models significantly ups the ante. In this section, we illustrate connectionism’s hunger for energy by estimating the total carbon footprint created in the training of the GPT-3<sup>56</sup> and PaLM foundation models [17,270].

Let us start with the eminent GPT-3 model. Unfortunately, the total energy cost is not stated in Brown *et al.* [17] but we can make a ballpark estimate using information from that work. GPT-3 was trained on a high-performance computing cluster containing  $N = 10\,000$  NVIDIA V100 chips, and required a total  $\Sigma = 3.14 \times 10^{23}$  FLOPs to train to completion [17]. A single V100 has a throughput of  $C = 2.8 \times 10^{13}$  FLOPs for half-precision floats, and so we can estimate GPT-3’s total training time in datacentre-seconds as

$$\frac{\Sigma}{C \cdot N} = \frac{3.14 \times 10^{23}}{2.8 \times 10^{12} \cdot 10^4} = 1.12 \times 10^6 \text{ s,}$$

which is approximately 311 h. We know the thermal design power of a single V100 chip is 300 W and so we can safely assume a lower bound on the datacentre power usage as 3000 kW. Therefore, we estimate the total power consumed while training GPT-3 as

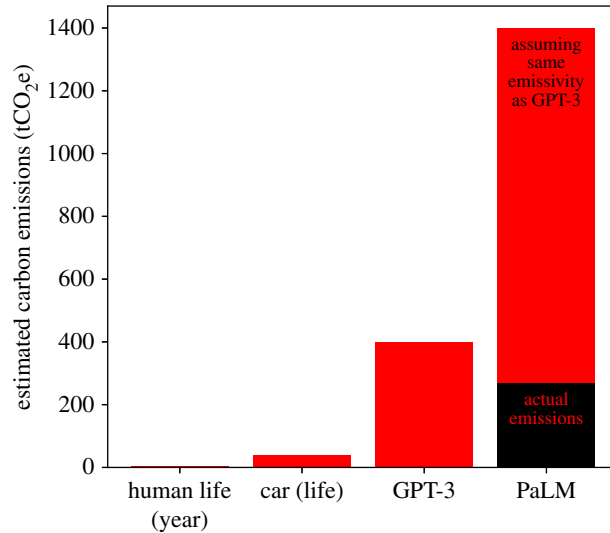
$$3000 \cdot 311 = 933\,000 \text{ kWh.}$$

The emissions per kWh of the datacentre where GPT-3 was trained is 0.429 kg CO<sub>2</sub>e kWh<sup>-1</sup> [321], leaving us with a total emission of around 400 000 kg CO<sub>2</sub>e.<sup>57</sup>

However, GPT-3 is already years old; so we will also estimate the energy used when training Google’s state-of-the-art ‘PaLM’ foundation model. Chowdhery *et al.* [270] state: ‘We trained PaLM-540B on 6144 TPU v4 chips for 1200 hours and 3072 TPU v4 chips for 336 hours including some downtime and repeated steps... [We found a] 378.5 W measured system power per TPU v4 chip...’ We can therefore

<sup>56</sup>We would compare GPT-4, but OpenAI has neglected to disclose any information regarding the training routine of the network in their ‘Technical Report’ [26].

<sup>57</sup>We must keep in mind that this estimate is a lower limit. We do not include CPU power, cooling or any other overheads in our calculation, never mind the cost to do a full hyperparameter sweep!



**Figure 31.** Here we contextualize the huge carbon footprints generated when training foundation models. The average person's yearly carbon footprint is estimated as 4750 kg CO<sub>2</sub>e using data from Friedlingstein *et al.* [322], and the car lifetime emissions is 38 504 kg CO<sub>2</sub>e assuming a Mercedes-Benz C 300 d model [323].

calculate PaLM's total energy usage as

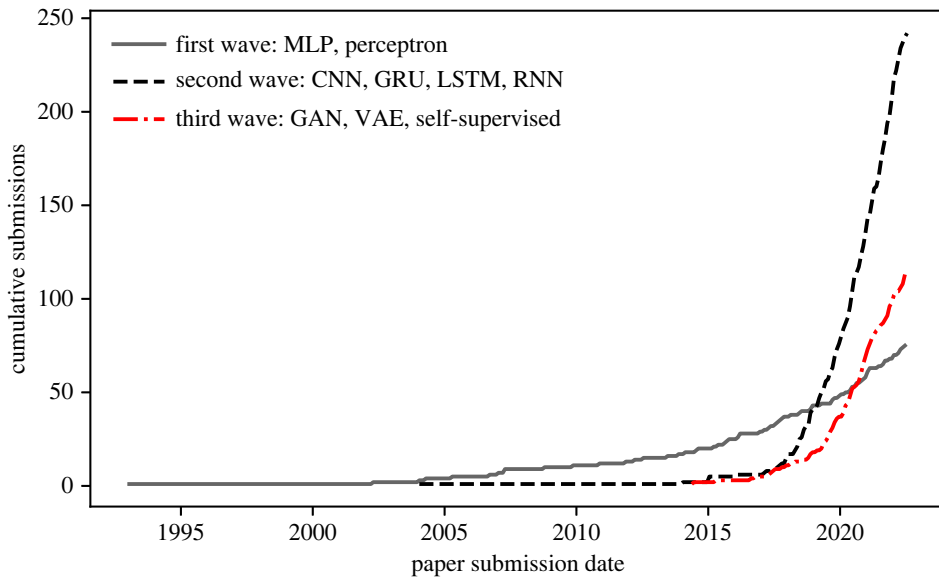
$$378.5 \cdot (6144 \cdot 1200 + 3072 \cdot 336) \approx 3\,180\,000 \text{ kWh.}$$

If PaLM was trained on the same datacentre as GPT-3 (i.e. at an emissivity of 0.429 kg CO<sub>2</sub>e kWh<sup>-1</sup>), it would have emitted a staggering 1 400 000 kg CO<sub>2</sub>e—quadruple the average person's lifetime carbon footprint [322] and approaching the annual emission of some small countries. Luckily, the datacentre that PaLM was trained on was far greener than that used by OpenAI, and PaLM actually produced approximately 270 000 kg CO<sub>2</sub>e [270], although this is still rather large. We contextualize our calculated footprints visually in figure 31.

PaLM's contribution to figure 31 demonstrates the importance of choosing and using datacentres that run on clean energy sources when training deep learning models and make efficient use of heat output (e.g. through recovery systems). Besides this, researchers can also take care when optimizing their neural network models to reduce their carbon footprint. For instance by choosing hyperparameters through a more efficient manual or randomized search, instead of via a brute force method [324]. As stated in Strubell *et al.* [325] researchers can also combat redundant retraining of models (and thus unnecessary energy usage) by ensuring that fully trained models, data and code are released under an open licence. The publishing of a fully trained model's energy usage, computation requirements and carbon footprint also allows downstream researchers to determine whether replication of a work is economically and environmentally viable. Calculating one's energy usage in the spirit of openness does not have to be difficult: we have been using the excellent and user-friendly 'Machine Learning CO<sub>2</sub> Impact Calculator' in our own work to calculate and publish the carbon footprint of our models [326]. A recommendation of this review is that an environmental impact statement should become standard practice in journal articles, conference presentations and proceedings when deep learning models (or any high-performance computing (HPC)-heavy research for that matter) is used.

## 11. Final comments, or how we learnt to stop worrying and love astronomy's Big Data Era

To repeat our introductory statement: in every field that deep learning has infiltrated we have seen a reduction in the use of specialist knowledge, to be replaced with knowledge automatically derived from data. We have already seen this process play out in many disparate fields from computer Go [15], to protein folding [16], to natural language processing [17], to computer vision [18]. This process is already well known within the deep learning community as 'The Bitter Lesson,' a precept that is summarized by the quote:



**Figure 32.** Here we see the number of arXiv:astro-ph submissions whose titles or abstracts match the terms given in the legend. We can see three distinct ‘waves’. The first corresponds to studies that use MLPs (§§2.1–3), the second corresponds to studies that use ‘deep learning’ methods that ingest raw data (§§4.1–5) and the third corresponds to studies that use generative or self-supervised models (§§6–8). The raw data are in the public domain, and are available at <https://www.kaggle.com/Cornell-University/arxiv>.

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. [91]

There is no reason to believe that astronomy is fundamentally different. Indeed, within this review we have seen a narrative pointing to this conclusion (figure 32). Initial work on MLPs within astronomy required manually selected emergent properties as input (e.g. [53,75]). With the advent of CNNs and RNNs, these manually selected inputs gave way to raw data ingestion (e.g. [131,155]). Now we are seeing the removal of human supervision altogether with deep learning methods inferring labels and knowledge directly from the data (e.g. [170,177]). Ultimately, if astronomy follows in the footsteps of other applied deep learning fields, we will see the removal of expertly crafted deep learning models, to be replaced with fine-tuned versions of an all-encompassing ‘foundation’ model [173]. This process is by no means a bad thing; the removal of human bias in the astronomical discovery process allows us to find ‘unknown unknowns’ through serendipity [169,261]. Likewise, the ability to leverage data allows us to directly generate and interrogate realistic yet synthetic observations, sidestepping the need for an expensive and fragile classical simulation [13,239].

Astronomy’s relative data wealth gives us the opportunity to form a symbiotic relationship with the cutting edge of deep learning research, an increasingly data hungry field [92,280]. Many ultra-large datasets in machine learning are proprietary, and so the astronomical community has the opportunity to step in and provide a high-quality multi-modal public dataset. In turn, this dataset could be used to train an astronomical ‘foundation’ model that can be used for state-of-the-art downstream tasks (such as astronomical simulation, see §9.3.3). Finally, following recent developments in connectionism [17,273] most astronomers lack the resources to train models on the cutting edge of the field. If astronomy is to have any chance of keeping up with the Big Tech goliaths, we must follow the examples of EleutherAI and HuggingFace and pool our resources in a grassroots-style open source fashion (§9). We leave this as a challenge for the community.

**Data accessibility.** This article has no additional data.

**Authors’ contributions.** M.J.S.: conceptualization, writing—original draft, writing—review and editing; J.E.G.: writing—original draft, writing—review and editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

**Conflict of interest declaration.** We declare we have no competing interests.

**Funding.** M.J.S. acknowledges support from the Alan Turing Institute by way of the Turing Enrichment Scheme. J.E.G. acknowledges funding from the Royal Society and the Science and Technology Facilities Council.

**Acknowledgements.** The authors would like to thank Hans-Martin Adorf for providing access to his manuscripts. The galaxy icon shown in figure 16 is by Agata Kuczińska and is reproduced here under the CC-BY-4.0 licence. We



would like to thank Connor Stone, Micah Bowels and the anonymous reviewers for their helpful comments and suggestions on the first draft of this manuscript.

**Disclaimer.** This study is an adaptation of work presented in chapters 1 and 5 of M.J.S.'s PhD thesis [327]. GPT-4 wrote the abstract.

## References

- Leibniz GW. 1666 Dissertation on the art of combinations. In *Philosophical papers and letters* (ed. LE Loemker), pp. 73–84. Dordrecht, The Netherlands: Springer.
- Fidora A, Sierra C. 2011 *Ramon Llull: from the Ars Magna to artificial intelligence*. Barcelona, Spain: Artificial Intelligence Research Inst.
- Gray J. 2016 'Let us Calculate!' Leibniz, Llull, and the computational imagination. See <https://publicdomainreview.org/essay/let-us-calculateleibniz-llull-and-the-computational-imagination>.
- Turing AM. 1950 I.—Computing machinery and intelligence. *Mind* **LIX**, 433–460. (doi:10.1093/mind/LIX.236.433)
- Moor J. 2006 The Dartmouth College Artificial Intelligence Conference: the next fifty years. *AI Mag.* **27**, 87.
- Bostrom N. 2014 *Superintelligence: paths, dangers, strategies*. Oxford, UK: Oxford University Press.
- Mitchell M. 2019 *Artificial intelligence: a guide for thinking humans*. London, UK: Penguin Books.
- Lahav O. 1994 Artificial neural networks as non-linear extensions of statistical methods in astronomy. *Vistas in Astron.* **38**, 251–256. (doi:10.1016/0083-6656(94)90034-5)
- Tagliaferri R, Longo G, Andreon S, Capozziello S, Donalek C, Giordano G. 2003 Neural networks for photometric redshifts evaluation. In *Neural nets* (eds B Apolloni, M Marinaro, R Tagliaferri), pp. 226–234. Berlin, Germany: Springer.
- Firth AE, Lahav O, Somerville RS. 2003 Estimating photometric redshifts with artificial neural networks. *Mon. Not. R. Astron. Soc.* **339**, 1195–1202. (doi:10.1046/j.1365-8711.2003.06271.x)
- Hayat MA, Stein G, Harrington P, Lukić Z, Mustafa M. 2021 Self-supervised representation learning for astronomical images. *Astrophys. J. Lett.* **911**, L33. (doi:10.3847/2041-8213/abf2c7)
- Bretonnière H *et al.* 2022 Euclid preparation – XIII. Forecasts for galaxy morphology with the Euclid Survey using deep generative models. *Astron. Astrophys.* **657**, A90.
- Smith MJ, Geach JE, Jackson RA, Arora N, Stone C, Courteau S. 2022 Realistic galaxy image simulation via score-based generative models. *Mon. Not. R. Astron. Soc.* **511**, 1808–1818. (doi:10.1093/mnras/stac130)
- LeCun Y. 2017 *My take on Ali Rahimi's 'Test of Time' award talk at NIPS*. Facebook. See [https://www2.isye.gatech.edu/~tzhao80/Yann\\_Response.pdf](https://www2.isye.gatech.edu/~tzhao80/Yann_Response.pdf).
- Silver D *et al.* 2016 Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489. (doi:10.1038/nature16961)
- Jumper J *et al.* 2021 Highly accurate protein structure prediction with AlphaFold. *Nature* **596**, 583–589. (doi:10.1038/s41586-021-03819-2)
- Brown T *et al.* 2020 Language models are few-shot learners. In *Advances in neural information processing systems* (eds H Larochelle, M Ranzato, R Hadsell, M Balcan, H Lin), vol. 33, pp. 1877–1901. Red Hook, NY: Curran Associates, Inc.
- Dosovitskiy A *et al.* 2020 An image is worth 16 × 16 words: transformers for image recognition at scale. (<http://arxiv.org/abs/2010.11929>)
- Zhang Y, Zhao Y. 2015 Astronomy in the big data era. *Data Sci. J.* **14**, 11. (doi:10.5334/dsj-2015-011)
- Springel V *et al.* 2018 First results from the IllustrisTNG simulations: matter and galaxy clustering. *Mon. Not. R. Astron. Soc.* **475**, 676–698. (doi:10.1093/mnras/stx3304)
- Vogelsberger M, Marinacci F, Torrey P, Puchwein E. 2020 Cosmological simulations of galaxy formation. *Nat. Rev. Phys.* **2**, 42–66. (doi:10.1038/s42254-019-0127-2)
- Angulo RE, Hahn O. 2022 Large-scale dark matter simulations. *Living Rev. Comput. Astrophys.* **8**, 1–200. (doi:10.1007/s41115-021-00013-z)
- Miller AS. 1993 A review of neural network applications in astronomy. *Vistas in Astron.* **36**, 141–161. (doi:10.1016/0083-6656(93)90118-4)
- Ball NM, Brunner RJ. 2010 Data mining and machine learning in astronomy. *Int. J. Modern Phys. D* **19**, 1049–1106. (doi:10.1142/S0218271810017160)
- Huertas-Company M, Lanusse F. 2023 The Dawes review 10: the impact of deep learning for the analysis of galaxy surveys. *Publ. Astron. Soc. Australia* **40**, e001. (doi:10.1017/pasa.2022.55)
- OpenAI. 2023 GPT-4 Technical Report. See <https://openai.com/research/gpt-4>.
- Bubeck S *et al.* 2023 Sparks of artificial general intelligence: early experiments with GPT-4. (<http://arxiv.org/abs/2303.12712>)
- McCulloch W, Pitts W. 1943 A logical calculus of ideas immanent in nervous activity. *Bull. Math. Biophys.* **5**, 127–147. (doi:10.1007/BF02478259)
- Rosenblatt F. 1958 The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **65**, 65–386. (doi:10.1037/h0042519)
- Hebb DO. 1949 *The organization of behavior: a neuropsychological theory*. New York, NY: Wiley.
- Minsky M, Papert S. 1969 *Perceptrons: an introduction to computational geometry*. Cambridge, MA: MIT Press.
- Olazaran M. 1996 A sociological study of the official history of the perceptrons controversy. *Soc. Stud. Sci.* **26**(3), 611–659. (doi:10.1177/030631296026003005)
- Metz C. 2021 *Genius makers: the mavericks who brought AI to Google, Facebook, and the world*. New York, NY: Penguin Random House.
- Ivakhnenko A, Lapa V. 1965 *Cybernetic predicting devices*. USSR: CCM Information Corporation.
- Ivakhnenko A. 1971 Polynomial theory of complex systems. *IEEE Trans. Syst. Man Cybern.* **SMC-1**, 364–378. (doi:10.1109/TSMC.1971.4308320)
- Rosenblatt F. 1962 *Principles of neurodynamics: perceptrons and the theory of brain mechanisms*. Cornell Aeronautical Laboratory. Report no. VG-1196-G-8. Spartan Books.
- Cybenko G. 1989 Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst. (MCSS)* **2**, 303–314. (doi:10.1007/BF02551274)
- Hornik K, Tinchcombe M, White H. 1991 Approximation capabilities of multilayer feedforward networks. *Neural Netw.* **4**, 251–257. (doi:10.1016/0893-6080(91)90009-T)
- Lu Z, Pu H, Wang F, Hu Z, Wang L. 2017 The expressive power of neural networks: a view from the width. In *Advances in neural information processing systems* (eds I Guyon, UV Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, R Garnett), vol. 30, pp. 6232–6240. Red Hook, NY: Curran Associates, Inc.
- Linnainmaa S. 1976 Taylor expansion of the accumulated rounding error. *BIT* **16**, 146–160. (doi:10.1007/BF01931367)
- Verbois PJ. 1981 Applications of advances in nonlinear sensitivity analysis. In *Proc. of the 10th IFIP Conf., NYC, 31 August–4 September*, pp. 762–770.
- Rumelhart DE, Hinton GE, Williams RJ. 1986 Learning representations by back-propagating errors. *Nature* **323**, 533–536. (doi:10.1038/323533a0)
- Linnainmaa S. 1970 The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. Master's thesis, The University of Helsinki, Finland. [In Finnish.]
- Schmidhuber J. 2014 Deep learning in neural networks: an overview. (<http://arxiv.org/abs/1404.7828>).
- Baydin AG, Pearlmutter BA, Radul AA, Siskind JM. 2018 Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.* **18**, 1–43.
- Fukushima K. 1980 Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybern.* **36**, 193–202. (doi:10.1007/BF00344251)



47. Nair V, Hinton GE. 2010 Rectified linear units improve restricted Boltzmann machines. In *Proc. of the 27th Int. Conf. on Machine Learning ICM'10, Haifa, Israel, 21–24 June*, pp. 807–814. Madison, WI: Omnipress.
48. Clevert D, Unterthiner T, Hochreiter S. 2016 Fast and accurate deep network learning by exponential linear units (ELUs). In *4th Int. Conf. on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2–4 May*. (<https://arxiv.org/abs/1511.07289>)
49. Ramachandran P, Zoph B, Le QV. 2017 Searching for activation functions. (<http://arxiv.org/abs/1710.05941>).
50. Misra D. 2019 Mish: a self regularized non-monotonic activation function. (<http://arxiv.org/abs/1908.08681>).
51. Rappaport B, Anderson K. 1988 Automated galaxy recognition. In *European Southern Observatory Conf. and Workshop Proc., Garching, Germany, 12–14 October*, vol. 28, pp. 233–238.
52. Adorf HM, Johnston MD. 1988 Artificial neural nets in astronomy. In *Arbeitspapier der Gesellschaft für Mathematik und Datenverarbeitung*, vol. 329.
53. Odewahn SC, Stockwell EB, Pennington RL, Humphreys RM, Zumach WA. 1992 Automated star/galaxy discrimination with neural networks. *Astron. J.* **103**, 318. (doi:10.1086/116063)
54. Pennington RL, Humphreys RM, Odewahn SC, Zumach W, Thurmes PM. 1993 The automated plate scanner catalog of the palomar sky survey. I. Scanning parameters and procedures on JSTOR. *Publ. Astron. Soc. Pac.* **105**, 521–526. (doi:10.1086/133186)
55. Odewahn SC, Humphreys RM, Aldering G, Thurmes P. 1993 Star-galaxy separation with a neural network. ii. multiple Schmidt plate fields. *Publ. Astron. Soc. Pac.* **105**, 1354. (doi:10.1086/133317)
56. Bazell D, Peng Y. 1998 A comparison of neural network algorithms and preprocessing methods for star-galaxy discrimination. *Astrophys. J. Suppl. Ser.* **116**, 47–55. (doi:10.1086/313098)
57. Bertin E, Arnouts S. 1996 SExtractor: software for source extraction. *Astron. Astrophys. Suppl. Ser.* **117**, 393–404. (doi:10.1051/aas:1996164)
58. Andreon S, Gargiulo G, Longo G, Tagliaferri R, Capuano N. 2000 Wide field imaging—I. Applications of neural networks to object detection and star/galaxy classification. *Mon. Not. R. Astron. Soc.* **319**, 700–716. (doi:10.1046/j.1365-8711.2000.03700.x)
59. Storrie-Lombardi MC, Lahav O. 1992 Morphological classification of galaxies by artificial neural networks. *Mon. Not. R. Astron. Soc.* **259**, 8P–12P. (doi:10.1093/mnras/259.1.8P)
60. Lahav O *et al.* 1995 Galaxies, human eyes, and artificial neural networks. *Science* **267**, 859–862. (doi:10.1126/science.267.5199.859)
61. Lahav O, Nairn A, Sodr e Jr L, Storrie-Lombardi MC. 1996 Neural computation as a tool for galaxy classification: methods and examples. *Mon. Not. R. Astron. Soc.* **283**, 207–221. (doi:10.1093/mnras/283.1.207)
62. Naim A, Lahav O. 1995 Automated morphological classification of APM galaxies by supervised artificial neural networks. *Mon. Not. R. Astron. Soc.* **275**, 567–590. (doi:10.1093/mnras/275.3.567)
63. Naim A *et al.* 1995 A comparative study of morphological classifications of APM galaxies. *Mon. Not. R. Astron. Soc.* **274**, 1107–1125.
64. Odewahn SC, Windhorst RA, Driver SP, Keel WC. 1996 ADS. *Astrophys. J. Lett.* **472**, L13. (doi:10.1086/310345)
65. Ball NM, Loveday J, Fukugita M, Nakamura O, Okamura S, Brinkmann J, Brunner RJ. 2004 Galaxy types in the Sloan Digital Sky Survey using supervised artificial neural networks. *Mon. Not. R. Astron. Soc.* **348**, 1038–1046. (doi:10.1111/j.1365-2966.2004.07429.x)
66. von Hippel T, Storrie-Lombardi LJ, Storrie-Lombardi MC, Irwin MJ. 1994 Automated classification of stellar spectra—I. Initial results with artificial neural networks. *Mon. Not. R. Astron. Soc.* **269**, 97–104. (doi:10.1093/mnras/269.1.97)
67. Klusch M, Napiwotzki R. 1993 HNS: a hybrid neural system and its use for the classification of stars. *Astron. Astrophys.* **276**, 309–319.
68. Chon MC. 1998 Muon physics and neural network event classifier for the Sudbury Neutrino Observatory. PhD thesis, University of Guelph, Canada.
69. Carballo R, Cofi o AS, Gonz alez-Serrano JJ. 2004 Selection of quasar candidates from combined radio and optical surveys using neural networks. *Mon. Not. R. Astron. Soc.* **353**, 211–220. (doi:10.1111/j.1365-2966.2004.08056.x)
70. Claeskens JF, Smette A, Vandenbulcke L, Surdej J. 2006 Identification and redshift determination of quasi-stellar objects with medium-band photometry: application to Gaia. *Mon. Not. R. Astron. Soc.* **367**, 879–904. (doi:10.1111/j.1365-2966.2006.10024.x)
71. Carballo R, Gonz alez-Serrano JJ, Benn CR, Jim enez-Luj an F. 2008 Use of neural networks for the identification of new  $z \geq 3.6$  QSOs from FIRST–SDSS DR5. *Mon. Not. R. Astron. Soc.* **391**, 369–382. (doi:10.1111/j.1365-2966.2008.13896.x)
72. White RL *et al.* 2000 The FIRST bright quasar survey. II. 60 nights and 1200 spectra later. *Astrophys. J. Suppl. Ser.* **126**, 133–207. (doi:10.1086/313300)
73. Kessler R *et al.* 2010 Results from the supernova photometric classification challenge. *Publ. Astron. Soc. Pac.* **122**, 1415–1431. (doi:10.1086/657607)
74. Karpenka NV, Feroz F, Hobson MP. 2013 A simple and robust method for automated photometric classification of supernovae using neural networks. *Mon. Not. R. Astron. Soc.* **429**, 1278–1285. (doi:10.1093/mnras/sts412)
75. Angel J, Wizinowich P, Lloyd-Hart M, Sandler D. 1990 Adaptive optics for array telescopes using neural-network techniques. *Nature* **348**, 221–224. (doi:10.1038/348221a0)
76. Sandler DG, Barrett TK, Palmer DA, Fugate RQ, Wild WJ. 1991 Use of a neural network to control an adaptive optics system for an astronomical telescope. *Nature* **351**, 300–302. (doi:10.1038/351300a0)
77. Lloyd-Hart M, Wizinowich P, McLeod B, Wittman D, Colucci D, Dekany R, McCarthy D, Angel JRP, Sandler D. 1992 First results of an on-line adaptive optics system with atmospheric wavefront sensing by an artificial neural network. *Astrophys. J. Lett.* **390**, L41. (doi:10.1086/186367)
78. Vanzella E *et al.* 2004 Photometric redshifts with the multilayer perceptron neural network: application to the HDF-S and SDSS. *Astron. Astrophys.* **423**, 761–776. (doi:10.1051/0004-6361:20040176)
79. Collister AA, Lahav O. 2004 ANNz: estimating photometric redshifts using artificial neural networks. *Publ. Astron. Soc. Pac.* **116**, 345–351. (doi:10.1086/383254)
80. Stoughton C *et al.* 2002 Sloan digital sky survey: early data release. *Astron. J.* **123**, 485–548. (doi:10.1086/324741)
81. York DG *et al.* 2000 The Sloan digital sky survey: technical summary. *Astron. J.* **120**, 1579–1587. (doi:10.1086/301513)
82. Koons HC, Gorney DJ. 1990 A sunspot maximum prediction using a neural network. *Eos Trans. Am. Geophys. Union* **71**, 677–688. (doi:10.1029/E0071018p00677-01)
83. Bailer-Jones CAL, Irwin M, Gilmore G, von Hippel T. 1997 Physical parametrization of stellar spectra: the neural network approach. *Mon. Not. R. Astron. Soc.* **292**, 157–166. (doi:10.1093/mnras/292.1.157)
84. Auld T, Bridges M, Hobson MP, Gull SF. 2007 Fast cosmological parameter estimation using neural networks. *Mon. Not. R. Astron. Soc.: Lett.* **376**, L11–L15. (doi:10.1111/j.1745-3933.2006.00276.x)
85. Auld T, Bridges M, Hobson MP. 2008 cosmonet: fast cosmological parameter estimation in non-flat models using neural networks. *Mon. Not. R. Astron. Soc.* **387**, 1575–1582. (doi:10.1111/j.1365-2966.2008.13279.x)
86. N rsgaard-Nielsen HU, J rgensen HE. 2008 Foreground removal from CMB temperature maps using an MLP neural network. *Astrophys. Space Sci.* **318**, 195–206. (doi:10.1007/s10509-008-9912-6)
87. Tolstikhin IO *et al.* 2021 MLP-Mixer: an all-MLP architecture for vision. In *Advances in neural information processing systems* (eds M Ranzato, A Beygelzimer, Y Dauphin, P Liang, JW Vaughan), vol. 34, pp. 24261–24272. Curran Associates, Inc.
88. Touvron H *et al.* 2021 ResMLP: feedforward networks for image classification with data-efficient training. (<http://arxiv.org/abs/2105.03404>).
89. Liu H, Dai Z, So DR, Le QV. 2021 Pay attention to MLPs. (<http://arxiv.org/abs/2105.08050>).
90. Melas-Kyriazi L. 2021 Do you even need attention? A stack of feed-forward layers does surprisingly well on ImageNet. (<http://arxiv.org/abs/2105.02723>).
91. Sutton R. 2019 The bitter lesson. See <http://incompleteideas.net/InclIdeas/BitterLesson.html>.
92. Branwen G. 2022 The scaling hypothesis. See <https://www.gwern.net/Scaling-hypothesis>.
93. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. 1989 Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1**, 541–551. (doi:10.1162/neco.1989.1.4.541)
94. Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B. 2021 High-resolution image synthesis

- with latent diffusion models. (<http://arxiv.org/abs/2112.10752>).
95. Lin M, Chen Q, Yan S. 2013 Network in network. (<http://arxiv.org/abs/1312.4400>).
  96. Crawford K. 2015 *Bright spiral galaxy M81*. See <https://apod.nasa.gov/apod/ap151017.html> (accessed 16 July 2020).
  97. Werbos PJ. 1990 Backpropagation through time: what it does and how to do it. *Proc. IEEE* **78**, 1550–1560. (doi:10.1109/5.58337)
  98. Hochreiter S. 1991 Untersuchungen zu dynamischen neuronalen Netzen (investigations on dynamic neural networks). Diploma thesis, The Technical University of Munich, Germany. [In German.]
  99. Bengio Y, Simard P, Frasconi P. 1994 Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **5**, 157–166. (doi:10.1109/72.279181)
  100. Frankle J, Carbin M. 2018 The lottery ticket hypothesis: finding sparse, trainable neural networks. (<http://arxiv.org/abs/1803.03635>).
  101. Glorot X, Bordes A, Bengio Y. 2011 Deep sparse rectifier neural networks. In *Proc. of the 14th Int. Conf. on Artificial Intelligence and Statistics*, vol. 15 (eds G Gordon, D Dunson, M Dudík), pp. 315–323. Fort Lauderdale, FL: Proceedings of Machine Learning Research.
  102. Oh K, Jung K. 2004 GPU implementation of neural networks. *Pattern Recognit.* **37**, 1311–1314. (doi:10.1016/j.patcog.2004.01.013)
  103. Steinkra D, Simard P, Buck I. 2005 Using GPUs for machine learning algorithms. In *Proc. of the 8th Int. Conf. on Document Analysis and Recognition ICDAR '05, Seoul, South Korea, 31 August–1 September*, pp. 1115–1119. IEEE Computer Society.
  104. Chellapilla K, Puri S, Simard P. 2006 High performance convolutional neural networks for document processing. In *Tenth Int. Workshop on Frontiers in Handwriting Recognition La Baule (France)* (ed. G Lorette). Université de Rennes 1 Suvisoft. (<http://www.suvisoft.com>).
  105. Raina R, Madhavan A, Ng AY. 2009 Large-scale deep unsupervised learning using graphics processors. In *Proc. of the 26th Annual Int. Conf. on Machine Learning ICML '09, Montreal, Canada, 14–18 June*, pp. 873–880. New York, NY: Association for Computing Machinery.
  106. Cireşan D, Meier U, Gambardella LM, Schmidhuber J. 2010 Deep, big, simple neural nets for handwritten digit recognition. *Neural Comput.* **22**, 3207–3220. (doi:10.1162/NECO\_a\_00052)
  107. Cireşan D, Meier U, Masci J, Gambardella LM, Schmidhuber J. 2011 Flexible, high performance convolutional neural networks for image classification. In *IJCAI 2011, Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011* (ed. T Walsh), pp. 1237–1242. Washington, DC: IJCAI/AAAI.
  108. Krizhevsky A, Sutskever I, Hinton GE. 2012 ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conf. on Neural Information Processing Systems 2012. Proc. of a meeting held 3–6 December 2012, Lake Tahoe, NV, USA* (eds PL Bartlett, FCN Pereira, CJ Burges, L Bottou, KQ Weinberger), pp. 1106–1114. Association for Computing Machinery.
  109. Russakovsky O *et al.* 2015 ImageNet large scale visual recognition challenge. *Int. J. Comput. Vis. (IJCV)* **115**, 211–252. (doi:10.1007/s11263-015-0816-y)
  110. Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R. 2014 Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958.
  111. Sevilla J, Heim L, Ho A, Besiroglu T, Hobbhahn M, Villalobos P. 2022 Compute trends across three eras of machine learning. (<http://arxiv.org/abs/2202.05924>).
  112. Hochreiter S, Schmidhuber J. 1997 Long short-term memory. *Neural Comput.* **9**, 1735–1780. (doi:10.1162/neco.1997.9.8.1735)
  113. Gers F, Schmidhuber J, Cummins F. 2000 Learning to forget: continual prediction with LSTM. *Neural Comput.* **12**, 2451–2471. (doi:10.1162/089976600300015015)
  114. Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. 2014 Learning phrase representations using RNN encoder-decoder for statistical machine translation. (<http://arxiv.org/abs/1406.1078>).
  115. Bayer J. 2015 Learning sequence representations. PhD thesis, Technical University Munich, Germany.
  116. Sutskever I, Vinyals O, Le QV. 2014 Sequence to sequence learning with neural networks. (<http://arxiv.org/abs/1409.3215>).
  117. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. 2017 Attention is all you need. (<http://arxiv.org/abs/1706.03762>).
  118. Srivastava RK, Greff K, Schmidhuber J. 2015 Highway networks. (<http://arxiv.org/abs/1505.00387>).
  119. He K, Zhang X, Ren S, Sun J. 2015 Deep residual learning for image recognition. (<http://arxiv.org/abs/1512.03385>).
  120. Ronneberger O, Fischer P, Brox T. 2015 U-Net: convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015 – 18th Int. Conf. Munich, Germany, 5–9 October 2015, Proceedings, Part III* vol. 9351. *Lecture Notes in Computer Science*, pp. 234–241. Berlin, Germany: Springer.
  121. Bahdanau D, Cho K, Bengio Y. 2014 Neural machine translation by jointly learning to align and translate. (<http://arxiv.org/abs/1409.0473>).
  122. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. 2019 Language models are unsupervised multitask learners. *OpenAI Whitepaper*. See [https://cdn.openai.com/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf).
  123. Devlin J, Chang M, Lee K, Toutanova K. 2019 BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, 2–7 June, vol. 1 (Long and Short Papers)*, pp. 4171–4186. Minneapolis, MN: Association for Computational Linguistics.
  124. Reed S *et al.* 2022 A generalist agent. *Trans. Mach. Learn. Res.* See <http://arxiv.org/abs/2205.06175>
  125. Gori M, Monfardini G, Scarselli F. 2005 A new model for learning in graph domains. In *Proc. 2005 IEEE Int. Joint Conf. on Neural Networks, Montreal, Canada, 31 July–4 August*, vol. 2, pp. 729–734. New York, NY: IEEE.
  126. Bruna J, Zaremba W, Szlam A, LeCun Y. 2013 Spectral networks and locally connected networks on graphs. (<http://arxiv.org/abs/1312.6203>).
  127. Kipf TN, Welling M. 2017 Semi-supervised classification with graph convolutional networks. In *Int. Conf. on Learning Representations*. (<https://openreview.net/forum?id=SJU4ayYgl>).
  128. Li Y, Tarlow D, Brockschmidt M, Zemel R. 2015 Gated graph sequence neural networks. (<http://arxiv.org/abs/1511.05493>).
  129. Zhu WW *et al.* 2014 Searching for pulsars using image pattern recognition. *Astrophys. J.* **781**, 117. (doi:10.1088/0004-637X/781/2/117)
  130. Hála P. 2014 Spectral classification using convolutional neural networks. (<http://arxiv.org/abs/1412.8341>).
  131. Dieleman S, Willett KW, Dambre J. 2015 Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Mon. Not. R. Astron. Soc.* **450**, 1441–1459. (doi:10.1093/mnras/stv632)
  132. Raddick MJ, Bracey G, Gay PL, Lintott CJ, Murray P, Schawinski K, Szalay AS, Vandenberg J. 2010 Galaxy Zoo: exploring the motivations of citizen science volunteers. *Astron. Educ. Rev.* **9**, 010103. (doi:10.3847/AER2009036)
  133. Huertas-Company M *et al.* 2015 A catalog of visual-like morphologies in the 5 CANDELS fields using deep learning. *Astrophys. J. Suppl. Ser.* **221**, 8. (doi:10.1088/0067-0049/221/1/8)
  134. Koekemoer AM *et al.* 2011 CANDELS: the cosmic assembly near-infrared deep extragalactic legacy survey—the Hubble Space Telescope observations, imaging data products, and mosaics. *Astrophys. J. Suppl. Ser.* **197**, 36. (doi:10.1088/0067-0049/197/2/36)
  135. Anjyan AK, Thorat K. 2017 Classifying radio galaxies with the convolutional neural network. *Astrophys. J. Suppl. Ser.* **230**, 20. (doi:10.3847/1538-4365/aa7333)
  136. Wilde J, Serjeant S, Bromley JM, Dickinson H, Koopmans LVE, Metcalf RB. 2022 Detecting gravitational lenses using machine learning: exploring interpretability and sensitivity to rare lensing configurations. *Mon. Not. R. Astron. Soc.* **512**, 3464–3479. (doi:10.1093/mnras/stac562)
  137. Zeiler MD, Fergus R. 2014 Visualizing and understanding convolutional networks. In *Computer Vision – ECCV 2014* (eds D Fleet, T Pajdla, B Schiele, T Tuytelaars), pp. 818–833. Cham, Switzerland: Springer.
  138. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. 2016 Grad-CAM: visual explanations from deep networks via gradient-based localization. (<http://arxiv.org/abs/1610.02391>).
  139. Akeret J, Chang C, Lucchi A, Refregier A. 2017 Radio frequency interference mitigation using deep convolutional neural networks. *Astron. Comput.* **18**, 35–39. (doi:10.1016/j.ascom.2017.01.002)

140. Berger P, Stein G. 2019 A volumetric deep convolutional neural network for simulation of mock dark matter halo catalogues. *Mon. Not. R. Astron. Soc.* **482**, 2861–2871. (doi:10.1093/mnras/sty2949)
141. Milletari F, Navab N, Ahmadi SA. 2016 V-Net: fully convolutional neural networks for volumetric medical image segmentation. (<http://arxiv.org/abs/1606.04797>).
142. Aragon-Calvo MA. 2019 Classifying the large-scale structure of the universe with deep neural networks. *Mon. Not. R. Astron. Soc.* **484**, 5771–5784. (doi:10.1093/mnras/stz393)
143. Hausen R, Robertson BE. 2020 Morpheus: a deep learning framework for the pixel-level analysis of astronomical image Data. *Astrophys. J. Suppl. Ser.* **248**, 20. (doi:10.3847/1538-4365/ab8868)
144. Lauritsen L, Dickinson H, Bromley J, Serjeant S, Lim CF, Gao ZK, Wang WH. 2021 Superresolving Herschel imaging: a proof of concept using deep neural networks. *Mon. Not. R. Astron. Soc.* **507**, 1546–1556. (doi:10.1093/mnras/stab2195)
145. Choma N *et al.* 2018 Graph neural networks for IceCube signal classification. (<http://arxiv.org/abs/1809.06166>).
146. Villanueva-Domingo P, Villaescusa-Navarro F, Genel S, Anglés-Alcázar D, Hernquist L, Marinacci F, Spergel DN, Vogelsberger M, Narayanan D. 2021 Weighing the milky way and andromeda with artificial intelligence. (<http://arxiv.org/abs/2111.14874>).
147. Villanueva-Domingo P *et al.* 2022 Inferring halo masses with graph neural networks. *Astrophys. J.* **935**, 30. (doi:10.3847/1538-4357/ac7aa3)
148. Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM. 2018 Dynamic graph CNN for learning on point clouds. (<http://arxiv.org/abs/1801.07829>)
149. Aussem A, Murtagh F, Sarazin M. 1994 Dynamical recurrent neural networks and pattern recognition methods for time series prediction: application to seeing and temperature forecasting in the context of ESO's VLT astronomical weather station. *Vistas Astron.* **38**, 357–374. (doi:10.1016/0083-6656(94)90047-7)
150. Wu JG, Lundstedt H. 1996 Prediction of geomagnetic storms from solar wind data using Elman recurrent neural networks. *Geophys. Res. Lett.* **23**, 319–322. (doi:10.1029/96GL00259)
151. Lundstedt H, Gleisner H, Wintoft P. 2002 Operational forecasts of the geomagnetic *Dst* index. *Geophys. Res. Lett.* **29**, 34–1–34–4. (doi:10.1029/2002GL016151)
152. Vassiliadis D, Klimas AJ, Valdivia JA, Baker DN. 2000 The nonlinear dynamics of space weather. *Adv. Space Res.* **26**, 197–207. (doi:10.1016/S0273-1177(99)01050-9)
153. Brodrick D, Taylor D, Diederich J. 2004 Recurrent neural networks for narrowband signal detection in the time-frequency domain. *Symp. Int. Astron. Union* **213**, 483–486. (doi:10.1017/S0074180900193751)
154. Elman JL. 1990 Finding structure in time. *Cogn. Sci.* **14**, 179–211. (doi:10.1207/s15516709cog1402\_1)
155. Charnock T, Moss A. 2017 Deep recurrent neural networks for supernovae classification. *Astrophys. J. Lett.* **837**, L28. (doi:10.3847/2041-8213/aa603d)
156. Naul B, Bloom JS, Pérez F, van der Walt S. 2018 A recurrent neural network for classification of unevenly sampled variable stars. *Nat. Astron.* **2**, 151–155. (doi:10.1038/s41550-017-0321-z)
157. Gonzalez CAG, Absil O, Van Droogenbroeck M. 2018 Supervised detection of exoplanets in high-contrast imaging sequences. *Astron. Astrophys.* **613**, A71. (doi:10.1051/0004-6361/201731961)
158. Carrasco-Davis R, Cabrera-Vives G, Förster F, Estévez PA, Huijse J, Protopapas P, Reyes I, Martínez-Palomera J, Donato C. 2019 Deep learning for image sequence classification of astronomical events. *Publ. Astron. Soc. Pac.* **131**, 108006. (doi:10.1088/1538-3873/aaef12)
159. Finke T, Krämer M, Manconi S. 2021 Classification of Fermi-LAT sources with deep learning using energy and time spectra. *Mon. Not. R. Astron. Soc.* **507**, 4061–4073. (doi:10.1093/mnras/stab2389)
160. Weddell SJ, Webb RY. 2008 Reservoir computing for prediction of the spatially-variant point spread function. *IEEE J. Sel. Top. Signal Process.* **2**, 624–634. (doi:10.1109/JSTSP.2008.2004218)
161. Jaeger H, Haas H. 2004 Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science* **304**, 78–80. (doi:10.1126/science.1091277)
162. Capizzi G, Napoli C, Paternò L. 2012 An innovative hybrid neuro-wavelet method for reconstruction of missing data in astronomical photometric surveys. In *Artificial intelligence and soft computing* (eds L Rutkowski, M Korytkowski, R Scherer, R Tadeusiewicz, LA Zadeh, JM Zurada), pp. 21–29. Berlin, Germany: Springer.
163. Shen H, George D, Huerta EA, Zhao Z. 2019 Denoising gravitational waves with enhanced deep recurrent denoising auto-encoders. In *ICASSP 2019 – 2019 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May*, pp. 3237–3241. New York, NY: IEEE.
164. Morningstar WR, Levasseur LP, Hezaveh YD, Blandford R, Marshall P, Putzky P, Rueter TD, Wechsler R, Welling M. 2019 Data-driven reconstruction of gravitationally lensed galaxies using recurrent inference machines. *Astrophys. J.* **883**, 14. (doi:10.3847/1538-4357/ab35d7)
165. Liu H, Liu C, Wang JTL, Wang H. 2019 Predicting solar flares using a long short-term memory network. *Astrophys. J.* **877**, 121. (doi:10.3847/1538-4357/ab1b3c)
166. Kügler SD, Gianniotis N, Polsterer KL. 2016 An explorative approach for inspecting Kepler data. *Mon. Not. R. Astron. Soc.* **455**, 4399–4405. (doi:10.1093/mnras/stv2604)
167. Smith MJ, Arora N, Stone C, Courteau S, Geach JE. 2021 Pix2Prof: fast extraction of sequential information from galaxy imagery via a deep natural language ‘captioning’ model. *Mon. Not. R. Astron. Soc.* **503**, 96–105. (doi:10.1093/mnras/stab424)
168. Parmar N, Vaswani A, Uszkoreit J, Kaiser L, Shazeer N, Ku A, Tran D. 2018 Image transformer. In *Proc. of the 35th Int. Conf. on Machine Learning* (eds J Dy, A Krause), vol. 80, pp. 4055–4064. Proceedings of Machine Learning Research.
169. Donoso-Oliva C, Becker I, Protopapas P, Cabrera-Vives G, Vishnu M, Vardhan H. 2023 ASTROMER – a transformer-based embedding for the representation of light curves. *Astron. Astrophys.* **670**, A54. (doi:10.1051/0004-6361/202243928)
170. Morvan M, Nikolaou N, Yip KH, Waldmann I. 2022 Don't pay attention to the noise: learning self-supervised representations of light curves with a denoising time series transformer. *arXiv*. See <http://arxiv.org/abs/2207.02777>.
171. Ricker GR *et al.* 2015 Transiting exoplanet survey satellite (TESS). *J. Astron. Telescopes Instrum. Syst.* **1**, 014003. (doi:10.1117/1.JATIS.1.1.014003)
172. Pan J, Ting YS, Yu J. 2022 Astroconformer: inferring surface gravity of stars from stellar light curves with transformer. (<http://arxiv.org/abs/2207.02787>).
173. Bommasani R *et al.* 2021 On the opportunities and risks of foundation models. (<http://arxiv.org/abs/2108.07258>).
174. Rumelhart DE, Hinton GE, Williams RJ. 1986 Learning internal representations by error propagation. In *Parallel distributed processing: explorations in the microstructure of cognition: vol. 1: Foundations* (eds DE Rumelhart, JL McClelland, PDP Research Group), ch. 4, pp. 318–362. Cambridge, MA: MIT Press.
175. Kingma DP, Welling M. 2013 Auto-encoding variational Bayes. (<http://arxiv.org/abs/1312.6114>).
176. Regier J, McAuliffe J, Prabhat. 2015 A deep generative model for astronomical images of galaxies. In *NIPS Workshop on Advances in Approximate Bayesian Inference*. See <https://regier.stat.lsa.umich.edu/assets/pdf/regier2015deep.pdf>.
177. Spindler A, Geach JE, Smith MJ. 2020 AstroVaDEr: astronomical variational deep embedder for unsupervised morphological classification of galaxies and synthetic image generation. *Mon. Not. R. Astron. Soc.* **502**, 985. (doi:10.1093/mnras/staa3670)
178. Dosovitskiy A, Brox T. 2016 Generating images with perceptual similarity metrics based on deep networks. (<http://arxiv.org/abs/1602.02644>).
179. Zhao S, Song J, Ermon S. 2017 Towards deeper understanding of variational autoencoding models. (<http://arxiv.org/abs/1702.08658>).
180. Oord A, Vinyals O, Kavukcuoglu K. 2017 Neural discrete representation learning. (<http://arxiv.org/abs/1711.00937>)
181. Vahtad A, Kautz J. 2020 NWE: a deep hierarchical variational autoencoder. *Adv. Neural Inf. Process. Syst.* **33**, 19 667–19 679.
182. Child R. 2020 Very deep VAEs generalize autoregressive models and can outperform them on images. (<http://arxiv.org/abs/2011.10650>).
183. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. 2014 Generative adversarial nets. In *Advances in neural information processing systems 27* (eds Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger), pp. 2672–2680. Red Hook, NY: Curran Associates, Inc.
184. Isola P, Zhu JY, Zhou T, Efros AA. 2016 Image-to-image translation with conditional

- adversarial networks. (<http://arxiv.org/abs/1611.07004>).
185. Mirza M, Osindero S. 2014 Conditional generative adversarial nets. (<http://arxiv.org/abs/1411.1784>).
  186. Brock A, Donahue J, Simonyan K. 2018 Large scale GAN training for high fidelity natural image synthesis. (<http://arxiv.org/abs/1809.11096>).
  187. Kang M, Zhu JY, Zhang R, Park J, Shechtman E, Paris S, Park T. 2023 Scaling up GANs for text-to-image synthesis. (<http://arxiv.org/abs/2303.05511>).
  188. Cheng J, Dong L, Lapata M. 2016 Long short-term memory-networks for machine reading. (<http://arxiv.org/abs/1601.06733>).
  189. Karras T, Laine S, Aila T. 2018 A style-based generator architecture for generative adversarial networks. (<http://arxiv.org/abs/1812.04948>).
  190. Ledig C *et al.* 2016 Photo-realistic single image super-resolution using a generative adversarial network. (<http://arxiv.org/abs/1609.04802>).
  191. Yu J, Lin Z, Yang J, Shen X, Lu X, Huang TS. 2018 Generative image inpainting with contextual attention. (<http://arxiv.org/abs/1801.07892>).
  192. Weng L. 2019 From GAN to WGAN. (<http://arxiv.org/abs/1904.08994>).
  193. Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S. 2015 Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. of the 32nd Int. Conf. on Machine Learning, Lille, France, 6–11 July* (eds F Bach, D Blei), vol. 37, pp. 2256–2265. Proceedings of Machine Learning Research.
  194. Ho J, Jain A, Abbeel P. 2020 Denoising diffusion probabilistic models. In *Advances in neural information processing systems* (eds H Larochelle, M Ranzato, R Hadsell, MF Balcan, H Lin), vol. 33, pp. 6840–6851. Red Hook, NY: Curran Associates, Inc.
  195. Hyvärinen A. 2005 Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.* **6**, 695–709.
  196. Vincent P. 2011 A connection between score matching and denoising autoencoders. *Neural Comput.* **23**, 1661–1674. (doi:10.1162/NECO\_a\_00142)
  197. Song Y, Ermon S. 2020 Improved techniques for training score-based generative models. In *Advances in neural information processing systems* (eds H Larochelle, M Ranzato, R Hadsell, MF Balcan, H Lin), vol. 33, pp. 12 438–12 448. Red Hook, NY: Curran Associates, Inc.
  198. Jolicoeur-Martineau A, Piché-Taillefer R, Combes RT, Mitliagkas I. 2020 Adversarial score matching and improved sampling for image generation. (<http://arxiv.org/abs/2009.05475>).
  199. Jolicoeur-Martineau A, Li K, Piché-Taillefer R, Kachman T, Mitliagkas I. 2021 Gotta go fast when generating data with score-based models. (<http://arxiv.org/abs/2105.14080>).
  200. Song Y, Sohl-Dickstein J, Kingma DP, Kumar A, Ermon S, Poole B. 2021 Score-based generative modeling through stochastic differential equations. In *Int. Conf. on Learning Representations*. See <https://openreview.net/forum?id=PxtIG12RRHS>.
  201. Nichol A, Dhariwal P. 2021 Improved denoising diffusion probabilistic models. (<http://arxiv.org/abs/2102.09672>).
  202. Dhariwal P, Nichol A. 2021 Diffusion models beat GANs on image synthesis. (<http://arxiv.org/abs/2105.05233>).
  203. Ramesh A, Dhariwal P, Nichol A, Chu C, Chen M. 2022 Hierarchical text-conditional image generation with CLIP latents. (<http://arxiv.org/abs/2204.06125>).
  204. Kadkhodaie Z, Simoncelli EP. 2020 Solving linear inverse problems using the prior implicit in a denoiser. (<http://arxiv.org/abs/2007.13640>).
  205. Saharia C, Ho J, Chan W, Salimans T, Fleet DJ, Norouzi M. 2021 Image super-resolution via iterative refinement. (<http://arxiv.org/abs/2104.07636>).
  206. Sasaki H, Willcocks CG, Breckon TP. 2021 UNIT-DDPM: Unpaired image translation with denoising diffusion probabilistic models. (<http://arxiv.org/abs/2104.05358>).
  207. Jayaram V, Thiekstun J. 2020 Source separation with deep generative priors. (<http://arxiv.org/abs/2002.07942>).
  208. Turner A. 2021 Diffusion models as a kind of VAE. See [https://angusturner.github.io/generative\\_models/2021/06/29/diffusion-probabilistic-models-1.html](https://angusturner.github.io/generative_models/2021/06/29/diffusion-probabilistic-models-1.html).
  209. Dieleman S. 2022 Diffusion models are autoencoders. See <https://benanne.github.io/2022/01/31/diffusion.html>.
  210. Song Y, Ermon S. 2019 Generative modeling by estimating gradients of the data distribution. In *Advances in neural information processing systems* (eds H Wallach, L Larochelle, A Beygelzimer, F d'Alché-Buc, E Fox, R Garnett), vol. 32. Red Hook, NY: Curran Associates, Inc.
  211. Luhman E, Luhman T. 2021 Knowledge distillation in iterative generative models for improved sampling speed. (<http://arxiv.org/abs/2101.02388>).
  212. Watson D, Chan W, Ho J, Norouzi M. 2022 Learning fast samplers for diffusion models by differentiating through sample quality. (<http://arxiv.org/abs/2202.05830>).
  213. Song J, Meng C, Ermon S. 2020 Denoising diffusion implicit models. (<http://arxiv.org/abs/2010.02502>).
  214. Chen T, Kornblith S, Norouzi M, Hinton G. 2020 A simple framework for contrastive learning of visual representations. (<http://arxiv.org/abs/2002.05709>).
  215. Chen T, Kornblith S, Swersky K, Norouzi M, Hinton G. 2020 Big self-supervised models are strong semi-supervised learners. (<http://arxiv.org/abs/2006.10029>).
  216. Grill JB *et al.* 2020 Bootstrap your own latent: a new approach to self-supervised learning. In *NIPS'20: Proc. of the 34th Int. Conf. on Neural Information Processing Systems, Online, 6–12 December*, pp. 21271–21284. Red Hook, NY: Curran Associates Inc.
  217. He K, Fan H, Wu Y, Xie S, Girshick R. 2019 Momentum contrast for unsupervised visual representation learning. (<http://arxiv.org/abs/1911.05722>).
  218. Chen X, Fan H, Girshick R, He K. 2020 Improved baselines with momentum contrastive learning. (<http://arxiv.org/abs/2003.04297>).
  219. Durkan C, Murray J, Papamakarios G. 2020 On contrastive learning for likelihood-free inference. (<http://arxiv.org/abs/2002.03712>).
  220. Hadsell R, Chopra S, LeCun Y. 2006 Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'06), Las Vegas, NV, 27–30 June*, vol. 2, pp. 1735–1742. New York NY: IEEE.
  221. Chechik G, Shama V, Shalit U, Bengio S. 2010 Large scale online learning of image similarity through ranking. *J. Mach. Learn. Res.* **11**, 1109–1135. (doi:10.1007/978-3-642-02172-5\_2)
  222. Sohn K. 2016 Improved deep metric learning with multi-class N-pair loss objective. In *Advances in neural information processing systems* (eds D Lee, M Sugiyama, U Luxburg, I Guyon, R Garnett), vol. 29. Red Hook, NY: Curran Associates, Inc.
  223. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S. 2017 GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in neural information processing systems* (eds I Guyon, U Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, R Garnett), vol. 30. Red Hook, NY: Curran Associates, Inc.
  224. Seitzer M. 2020 pytorch-fid: FID score for PyTorch, version 0.1.1. See <https://github.com/mseitzer/pytorch-fid>.
  225. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. 2016 Rethinking the inception architecture for computer vision. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 27–30 June*. New York, NY: IEEE.
  226. Regier J, Miller A, McAuliffe J, Adams R, Hoffman M, Lang D, Schlegel D, Prabhat Celeste: variational inference for a generative model of astronomical images. In *Int. Conf. on Machine Learning (ICML), Lille, France, 6–11 July*, pp. 2095–2103. Proceedings of Machine Learning Research.
  227. Serra-Ricart M, Calbet X, Garrido L, Gaitan V. 1993 Multidimensional statistical analysis using artificial neural networks: astronomical applications. *Astron. J.* **106**, 1685. (doi:10.1086/116758)
  228. Ravanbakhsh S, Lanusse F, Mandelbaum R, Schneider J, Poczos B. 2016 Enabling dark energy science with deep generative models of galaxy images. (<http://arxiv.org/abs/1609.05796>).
  229. Fussell L, Moews B. 2019 Forging new worlds: high-resolution synthetic galaxies with chained generative adversarial networks. *Mon. Not. R. Astron. Soc.* **485**, 3203–3214. (doi:10.1093/mnras/stz602)
  230. Holzschuh BJ, O'Riordan CM, Vegetti S, Rodriguez-Gomez V, Thurey N. 2022 Realistic galaxy images and improved robustness in machine learning tasks from generative modelling. *Mon. Not. R. Astron. Soc.* **515**, 652–677. (doi:10.1093/mnras/stac1188)
  231. Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A. 2016 Learning deep features for discriminative localization. In *Computer Vision and Pattern Recognition, Las Vegas, NV, 27–30 June* (ed. L O'Connor), pp. 2921–2929. Los Alamitos, CA: IEEE Computer Society.
  232. Weng L. 2018 Flow-based deep generative models. See <https://lilianweng.github.io/posts/2018-10-13-flow-models/>.



233. Germain M, Gregor K, Murray I, Larochelle H. 2015 MADE: masked autoencoder for distribution estimation. In *Int. Conf. on Machine Learning, Lille, France, 6–11 July*, pp. 881–889. PMLR.
234. Papamakarios G, Pavlakou T, Murray I. 2017 Masked autoregressive flow for density estimation. In *NIPS'17: Proc. of the 31st Int. Conf. on Neural Information Processing Systems, Long Beach, CA, 4–9 December*, pp. 2335–2344. Red Hook, NY: Curran Associates Inc.
235. Dey A *et al.* 2019 Overview of the DESI legacy imaging surveys. *Astron. J.* **157**, 168. (doi:10.3847/1538-3881/ab089d)
236. Abazajian KN *et al.* 2009 The seventh data release of the Sloan Digital Sky Survey. *Astrophys. J. Suppl. Ser.* **182**, 543–558. (doi:10.1088/0067-0049/182/2/543)
237. Wilman DJ, Zibetti S, Budavári T. 2010 A multiscale approach to environment and its influence on the colour distribution of galaxies. *Mon. Not. R. Astron. Soc.* **406**, 1701–1720. (doi:10.1111/j.1365-2966.2010.16845.x)
238. Stone C, Courteau S. 2019 The intrinsic scatter of the radial acceleration relation. *Astrophys. J.* **882**, 6. (doi:10.3847/1538-4357/ab3126)
239. Smith MJ, Geach JE. 2019 Generative deep fields: arbitrarily sized, random synthetic astronomical images through deep learning. *Mon. Not. R. Astron. Soc.* **490**, 4985–4990. (doi:10.1093/mnras/stz2886)
240. Jetchev N, Bergmann U, Vollgraf R. 2016 Texture synthesis with spatial generative adversarial networks. (http://arxiv.org/abs/1611.08207).
241. Rodriguez AC, Kacprzak T, Lucchi A, Amara A, Sgier R, Fluri J, Hofmann T, Refregier A. 2018 Fast cosmic web simulations with generative adversarial networks. *Comput. Astrophys. Cosmol.* **5**, 4. (doi:10.1186/s40668-018-0026-4)
242. Mustafa M, Bard D, Bhimiw J, Lukić Z, Al-Rfou R, Kratochvil JM. 2019 CosmoGAN: creating high-fidelity weak lensing convergence maps using generative adversarial networks. *Comput. Astrophys. Cosmol.* **6**, 1. (doi:10.1186/s40668-019-0029-9)
243. Remy B, Lanusse F, Ramzi Z, Liu J, Jeffrey N, Starck JL. 2020 Probabilistic mapping of dark matter by neural score matching. (http://arxiv.org/abs/2011.08271).
244. Remy B, Lanusse F, Jeffrey N, Liu J, Starck JL, Osato K, Schrabback T. 2023 Probabilistic mass-mapping with neural score estimation. *Astron. Astrophys.* **672**, A51. (doi:10.1051/0004-6361/202243054)
245. Liu J, Bird S, Matilla JM, Hill JC, Haiman Z, Madhavacheril MS, Petri A, Spergel DN. 2018 MassiveNuS: cosmological massive neutrino simulations. *J. Cosmol. Astropart. Phys.* **2018**, 049. (doi:10.1088/1475-7516/2018/03/049)
246. Scoville N *et al.* 2007 COSMOS: Hubble Space Telescope observations. *Astrophys. J. Suppl. Ser.* **172**, 38–45. (doi:10.1086/516580)
247. Schawinski K, Zhang C, Zhang H, Fowler L, Santhanam GK. 2017 Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit. *Mon. Not. R. Astron. Soc.* **467**, L110–L114. (doi:10.1093/mnras/slx008)
248. Stark D *et al.* 2018 psfgan: a generative adversarial network system for separating quasar point sources and host galaxy light. *Mon. Not. R. Astron. Soc.* **477**, 2513–2527. (doi:10.1093/mnras/sty764)
249. Reiman DM, Göhr BE. 2019 Deblending galaxy superpositions with branched generative adversarial networks. *Mon. Not. R. Astron. Soc.* **485**, 2617–2627. (doi:10.1093/mnras/stz575)
250. Adam A, Coogan A, Malkin N, Legin R, Perreault-Leveseur L, Hezaveh Y, Bengio Y. 2022 Posterior samples of source galaxies in strong gravitational lenses with score-based priors. (http://arxiv.org/abs/2211.03812).
251. Karchev K, Montel NA, Coogan A, Weniger C. 2022 Strong-lensing source reconstruction with denoising diffusion restoration models. (http://arxiv.org/abs/2211.04365).
252. Mudur N, Finkbeiner DP. 2022 Can denoising diffusion probabilistic models generate realistic astrophysical fields? (http://arxiv.org/abs/2211.12444)
253. Kawar B, Elad M, Ermon S, Song J. 2022 Denoising diffusion restoration models. (http://arxiv.org/abs/2211.04365).
254. Buncher B, Sharma AN, Carrasco-Kind M. 2021 Survey2Survey: a deep learning generative model approach for cross-survey image mapping. *Mon. Not. R. Astron. Soc.* **503**, 777–796. (doi:10.1093/mnras/stab294)
255. Arcelin B, Doux C, Aubourg E, Roucelle C, and LSST Dark Energy Science Collaboration. 2021 Deblending galaxies with variational autoencoders: a joint multiband, multi-instrument approach. *Mon. Not. R. Astron. Soc.* **500**, 531–547. (doi:10.1093/mnras/staa3062)
256. Calbet X, Mahoney T, Hammersley P, Garzon F, Selby M. 1993 ADS. *Galactic bulges: Proc. of the 153rd Symp. of the Int. Astronomical Union held in Ghent, Belgium, 17–22 August 1992*. (eds H DeJonghe, HJ Habing). International Astronomical Union. Symposium no. 153, p. 293. Dordrecht, The Netherlands: Kluwer Academic Publishers.
257. Graff P, Feroz F, Hobson MP, Lasenby A. 2014 ADS. *Mon. Not. R. Astron. Soc.* **441**, 1741–1759. (doi:10.1093/mnras/stu642)
258. Kitching TD *et al.* 2015 Image analysis for cosmology: shape measurement challenge review & results from the Mapping Dark Matter challenge. *Astron. Comput.* **10**, 9–21. (doi:10.1016/j.ascom.2014.12.004)
259. Yang T, Li X. 2015 An autoencoder of stellar spectra and its application in automatically estimating atmospheric parameters. *Mon. Not. R. Astron. Soc.* **452**, 158–168. (doi:10.1093/mnras/stv1210)
260. Tsang BTH, Schultz WC. 2019 Deep neural network classifier for variable stars with novelty detection capability. *Astrophys. J. Lett.* **877**, L14. (doi:10.3847/2041-8213/ab212c)
261. Sarmiento R, Huertas-Company M, Knapen JH, Sánchez SF, Sánchez HD, Drory N, Falcón-Barroso J. 2021 Capturing the physics of manga galaxies with self-supervised machine learning. *Astrophys. J.* **921**, 177. (doi:10.3847/1538-4357/ac1dac)
262. Bundy K *et al.* 2014 Overview of the SDSS-IV MaNGA survey: Mapping Nearby Galaxies at Apache Point Observatory. *Astrophys. J.* **798**, 7. (doi:10.1088/0004-637X/798/1/7)
263. Slijepcevic IV, Scaife AMM, Walmsley M, Bowles M. 2022 Learning useful representations for radio astronomy ‘in the wild’ with contrastive learning. (http://arxiv.org/abs/2207.08666).
264. Porter FAM. 2020 MiraiBest batched dataset. *Zenodo*. (doi:10.5281/zenodo.4288837)
265. Porby. 2022 Why I think strong general AI is coming soon. See <https://www.lesswrong.com/posts/K4urTDkBBtNuLivJx/why-i-think-strong-general-ai-is-coming-soon>.
266. Bo P. 2021 BlinkDL/RWKV-LM: 0.01. Version 0.01. See <https://doi.org/10.5281/zenodo.5196577>.
267. Kim J, Nguyen TD, Min S, Cho S, Lee M, Lee H, Hong S. 2022 Pure transformers are powerful graph learners. (http://arxiv.org/abs/2207.02505).
268. Alayrac J *et al.* 2022 Flamingo: a visual language model for few-shot learning. (http://arxiv.org/abs/2204.14198).
269. Wei J *et al.* 2022 Emergent abilities of large language models. (http://arxiv.org/abs/2206.07682).
270. Chowdhery A *et al.* 2022 PaLM: scaling language modeling with pathways. (http://arxiv.org/abs/2204.02311).
271. alumiug and Aggressive-Scheme-99. 2022 A reddit argument about flamingos. See [https://20/old.reddit.com/r/MachineLearning/comments/ue2ptk/r\\_flamingo\\_a\\_%20visual\\_language\\_model\\_for\\_fewshot/](https://20/old.reddit.com/r/MachineLearning/comments/ue2ptk/r_flamingo_a_%20visual_language_model_for_fewshot/).
272. Brock A, De S, Smith SL, Simonyan K. 2021 High-performance large-scale image recognition without normalization. (http://arxiv.org/abs/2102.06171).
273. Hoffmann J *et al.* 2022 Training compute-optimal large language models. (http://arxiv.org/abs/2203.15556).
274. Walmsley M, Slijepcevic IV, Bowles M, Scaife AMM. 2022 Towards galaxy foundation models with hybrid contrastive learning. (http://arxiv.org/abs/2206.11927).
275. Kaplan J *et al.* 2020 Scaling laws for neural language models. (http://arxiv.org/abs/2001.08361).
276. Thoppilan R *et al.* 2022 LaMDA: Language Models for Dialog Applications. *arXiv*. (doi:10.48550/arXiv.2201.08239)
277. Rae JW *et al.* 2021 Scaling Language Models: Methods, Analysis & Insights from Training Gopher. *arXiv*. (doi:10.48550/arXiv.2112.11446)
278. Smith S *et al.* 2022 Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, A Large-Scale Generative Language Model. *arXiv*. (doi:10.48550/arXiv.2201.11990)
279. Gao L *et al.* 2020 The Pile: an 800GB Dataset of diverse text for language modeling. (http://arxiv.org/abs/2101.00027)
280. Friel R. 2022 Chinchilla’s wild implications. See <https://www.alignmentforum.org/posts/6Fpvc8RRR29qLEWNH/chinchilla-s-wild-implications>.
281. Kaiser L, Gomez AN, Shazeer N, Vaswani A, Parmar N, Jones L, Uszkoreit J. 2017 One model to learn them all. (http://arxiv.org/abs/1706.05137)

282. Aghajanyan A *et al.* 2023 Scaling laws for generative mixed-modal language models. (<http://arxiv.org/abs/2301.03728>)
283. Ivezic Z *et al.* 2019 LSST: from science drivers to reference design and anticipated data products. *Astrophys. J.* **873**, 111.
284. Raymond ES. 1999 *The cathedral and the bazaar*, 1st edn. Sebastopol, CA: O'Reilly & Associates Inc.
285. Black S *et al.* 2022 GPT-NeoX-20B: an open-source autoregressive language model. (<http://arxiv.org/abs/2204.06745>).
286. Crowson K, Biderman S, Kornis D, Stander D, Hallahan E, Castriaco L, Raff E. 2022 VQGAN-CLIP: open domain image generation and editing with natural language guidance. (<http://arxiv.org/abs/2204.08583>).
287. BigScience Workshop *et al.* 2022 BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. (<http://arxiv.org/abs/2211.05100>).
288. Park JS, Popowski L, Cai CJ, Morris MR, Liang P, Bernstein MS. 2022 Social simulacra: creating populated prototypes for social computing systems. (<http://arxiv.org/abs/2208.04024>).
289. Aher G, Arriaga RI, Kalai AT. 2022 Using large language models to simulate multiple humans and replicate human subject studies. (<http://arxiv.org/abs/2208.10264>).
290. Radford A, Narasimhan K, Salimans T, Sutskever I. 2018 Improving language understanding by generative pre-training. *OpenAI Whitepaper*. See [https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf).
291. Knox WB, Stone P. 2008 TAMER: training an agent manually via evaluative reinforcement. In *2008 7th IEEE Int. Conf. on Development and Learning, Monterey, CA, 9–12 August*, pp. 292–297. New York, NY: IEEE.
292. Ziegler DM, Stiennon N, Wu J, Brown TB, Radford A, Amodei D, Christiano P, Irving G. 2019 Fine-tuning language models from human preferences. (<http://arxiv.org/abs/1909.08593>).
293. Touvron H *et al.* 2023 LLaMA: open and efficient foundation language models. (<http://arxiv.org/abs/2302.13971>).
294. Taori R, Gulrajani I, Zhang T, Dubois Y, Li X, Guestrin C, Liang P, Hashimoto TB. 2023 Stanford alpaca: an instruction-following LLaMA model. See [https://github.com/tatsu-lab/stanford\\_alpaca](https://github.com/tatsu-lab/stanford_alpaca) (accessed 14 April 2023).
295. Chiang WL *et al.* 2023 Vicuna: an open-source Chatbot impressing GPT-4 with 90%\* ChatGPT Quality. See <https://vicuna.lmsys.org> (accessed 14 April 2023).
296. Geng X, Gudibande A, Liu H, Wallace E, Abbeel P, Levine S, Song D. 2023 Koala: a dialogue model for academic research. Blog post. See <https://bair.berkeley.edu/blog/2023/04/03/koala/> (accessed 14 April 2023).
297. Beeching E, Belkada Y, Rasul K, Tunstall L, von Werra L, Rajani N, Lambert N. 2023 StackLLaMA: an RL fine-tuned LLaMA model for stack exchange question and answering. See <https://huggingface.co/blog/stackllama> (accessed 14 April 2023).
298. StabilityAI. 2023 Stable diffusion. See <https://github.com/Stability-AI/stablediffusion> (accessed 14 April 2023).
299. Sevilla J *et al.* 2022 Parameter, compute and data trends in machine learning. See [https://docs.google.com/spreadsheets/d/1AAlebjNsnJ\\_uKALHbXNfn3\\_YsT6sHXtCU0q7OIPuc4/](https://docs.google.com/spreadsheets/d/1AAlebjNsnJ_uKALHbXNfn3_YsT6sHXtCU0q7OIPuc4/).
300. Eloundou T, Manning S, Mishkin P, Rock D. 2023 GPTs are GPTs: an early look at the labor market impact potential of large language models. (<http://arxiv.org/abs/2206.07682>).
301. Brynjolfsson E. 2022 The Turing trap: the promise & peril of human-like artificial intelligence. (<http://arxiv.org/abs/2201.04200>)
302. Bowles M *et al.* 2023 Radio Galaxy Zoo EMU: towards a semantic radio galaxy morphology taxonomy. *Mon. Not. R. Astron. Soc.* **522**, stad1021. (doi:10.1093/mnras/stad1021)
303. Hinton GE, Roweis S. 2002 Stochastic neighbor embedding. In *Advances in neural information processing systems* (eds S Becker, S Thrun, K Obermayer), vol. 15. New York, NY: MIT Press.
304. van der Maaten L, Hinton G. 2008 Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605.
305. Park JS, O'Brien JC, Cai CJ, Morris MR, Liang P, Bernstein MS. 2023 Generative agents: interactive simulacra of human behavior. (<http://arxiv.org/abs/2304.03442>).
306. Nakajima Y. 2023 Task-driven autonomous agent utilizing GPT-4, Pinecone, and LangChain for diverse applications. See <https://yoheinakajima.com/task-driven-autonomous-agent-utilizing-gpt-4-pinecone-and-langchain-for-diverse-applications> (accessed 18 April 2023).
307. Liang Y *et al.* 2023 TaskMatrix.AI: completing tasks by connecting foundation models with millions of APIs. (<http://arxiv.org/abs/2303.16434>).
308. Saharia C *et al.* 2022 Photorealistic text-to-image diffusion models with deep language understanding. (<http://arxiv.org/abs/2205.11487>).
309. Ho J, Saharia C, Chan W, Fleet DJ, Norouzi M, Salimans T. 2021 Cascaded diffusion models for high fidelity image generation. (<http://arxiv.org/abs/2106.15282>).
310. Tay Y *et al.* 2021 Scale efficiently: insights from pre-training and fine-tuning transformers. (<http://arxiv.org/abs/2109.10686>).
311. Willett KW *et al.* 2013 Galaxy Zoo 2: detailed morphological classifications for 304 122 galaxies from the Sloan Digital Sky Survey. *Mon. Not. R. Astron. Soc.* **435**, 2835–2860. (doi:10.1093/mnras/stt1458)
312. Gal R, Alaluf Y, Atzmon Y, Patashnik O, Bermano AH, Chechik G, Cohen-Or D. 2022 An image is worth one word: personalizing text-to-image generation using textual inversion. (<http://arxiv.org/abs/2208.01618>).
313. Schirmer M, Diaz R, Holmberg K, Levenson NA, Winge C. 2013 A sample of Seyfert-2 galaxies with ultraluminous galaxy-wide narrow-line regions: quasar light echoes? *Astrophys. J.* **763**, 60. (doi:10.1088/0004-637X/763/1/60)
314. Kraus J. 1994 The tantalizing 'Wow!' signal - NRAO archives. See <https://www.nrao.edu/archives/items/show/3684> (accessed 11 April 2023).
315. Pang G, Shen C, Cao L, Hengel AVD. 2021 Deep learning for anomaly detection: a review. *ACM Comput. Surv.* **54**, 1–38. (doi:10.1145/3439950)
316. Wang M, Deng W. 2018 Deep visual domain adaptation: a survey. (<http://arxiv.org/abs/1802.03601>).
317. Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E, Le Q, Zhou D. 2022 Chain-of-thought prompting elicits reasoning in large language models. (<http://arxiv.org/abs/2201.11903>).
318. Wang H, Yeung DY. 2020 A survey on Bayesian deep learning. *ACM Comput. Surv.* **53**, 1–37. (doi:10.1145/3409383)
319. Tavare S, Balding DJ, Griffiths RC, Donnelly P. 1997 Inferring coalescence times from DNA sequence data. *Genetics* **145**, 505. (doi:10.1093/genetics/145.2.505)
320. Ras G, Xie N, van Gerven M, Doran D. 2020 Explainable deep learning: a field guide for the uninitiated. (<http://arxiv.org/abs/2004.14545>).
321. Patterson D, Gonzalez J, Le Q, Liang C, Munguia LM, Rothchild D, So D, Texier M, Dean J. 2021 Carbon emissions and large neural network training. (<http://arxiv.org/abs/2104.10350>).
322. Friedlingstein P *et al.* 2022 Global carbon budget 2021. *Earth Syst. Sci. Data* **14**, 1917–2005. (doi:10.5194/essd-14-1917-2022)
323. Buberger J, Kersten A, Kuder M, Eckerle R, Weyh T, Thiringer T. 2022 Total CO<sub>2</sub>-equivalent life-cycle emissions from commercially available passenger cars. *Renew. Sustain. Energy Rev.* **159**, 112158. (doi:10.1016/j.rser.2022.112158)
324. Bergstra J, Bengio Y. 2012 Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **13**, 281–305.
325. Strubell E, Ganesh A, McCallum A. 2019 Energy and policy considerations for deep learning in NLP. (<http://arxiv.org/abs/1906.02243>).
326. Lacoste A, Luccioni A, Schmidt V, Dandres T. 2019 Quantifying the carbon emissions of machine learning. (<http://arxiv.org/abs/1910.09700>).
327. Smith M. 2022 Using deep learning to explore ultra-large scale astronomical datasets. PhD thesis, University of Hertfordshire, UK.