Deep learning in electron microscopy

IOP Publishing Mach. Learn.: Sci. Technol. 2 (2021) 011004 https://doi.org/10.1088/2632-2153/abd614

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
5 October 2020

REVISED
3 November 2020

ACCEPTED FOR PUBLICATION
22 December 2020

PUBLISHED
31 March 2021

TOPICAL REVIEW

Deep learning in electron microscopy

Jeffrey M Ede

University of Warwick, Department of Physics, Coventry CV4 7AL, United Kingdom

E-mail: j.m.ede@warwick.ac.uk

Keywords: deep learning, electron microscopy, review

Abstract

Deep learning is transforming most areas of science and technology, including electron microscopy. This review paper offers a practical perspective aimed at developers with limited familiarity. For context, we review popular applications of deep learning in electron microscopy. Following, we discuss hardware and software needed to get started with deep learning and interface with electron microscopes. We then review neural network components, popular architectures, and their optimization. Finally, we discuss future directions of deep learning in electron microscopy.

1. Introduction

Following decades of exponential increases in computational capability [1] and widespread data availability [2, 3], scientists can routinely develop artificial neural networks [4–11] (ANNs) to enable new science and technology [12–17]. The resulting deep learning revolution [18, 19] has enabled superhuman performance in image classification [20–23], games [24–29], medical analysis [30, 31], relational reasoning [32], speech recognition [33, 34] and many other applications [35, 36]. This introduction focuses on deep learning in electron microscopy and is aimed at developers with limited familiarity. For context, we therefore review popular applications of deep learning in electron microscopy. We then review resources available to support researchers and outline electron microscopy. Finally, we review popular ANN architectures and their optimization, or 'training', and discuss future trends in artificial intelligence (AI) for electron microscopy.

Deep learning is motivated by universal approximator theorems [37–45], which state that sufficiently deep and wide [37, 40, 46] ANNs can approximate functions to arbitrary accuracy. It follows that ANNs can always match or surpass the performance of methods crafted by humans. In practice, deep neural networks (DNNs) reliably [47] learn to express [48–51] generalizable [52–59] models without a prior understanding of physics. As a result, deep learning is freeing physicists from a need to devise equations to model complicated phenomena [13, 14, 16, 60, 61]. Many modern ANNs have millions of parameters, so inference often takes tens of milliseconds on graphical processing units (GPUs) or other hardware accelerators [62]. It is therefore unusual to develop ANNs to approximate computationally efficient methods with exact solutions, such as the fast Fourier transform [63–65] (FFT). However, ANNs are able to leverage an understanding of physics to accelerate time-consuming or iterative calculations [66–69], improve accuracy of methods [30, 31, 70], and find solutions that are otherwise intractable [24, 71].

1.1. Improving signal-to-noise

A popular application of deep learning is to improve signal-to-noise [74, 75], for example, of medical electrical [76, 77], medical image [78–80], optical microscopy [81–84], and speech [85–88] signals. There are many traditional denoising algorithms that are not based on deep learning [89–91], including linear [92, 93] and non-linear [94–102] spatial domain filters, Wiener filters [103–105], non-linear [106–111] wavelet domain filters, curvelet transforms [112, 113], contourlet transforms [114, 115], hybrid algorithms [116–122] that operate in both spatial and transformed domains, and dictionary-based learning [123–127]. However, traditional denoising algorithms are limited by features (often laboriously) crafted by humans and cannot exploit domain-specific context. In perspective, they leverage an ever-increasingly accurate representation of physics to denoise signals. However, traditional algorithms are limited by the difficulty of

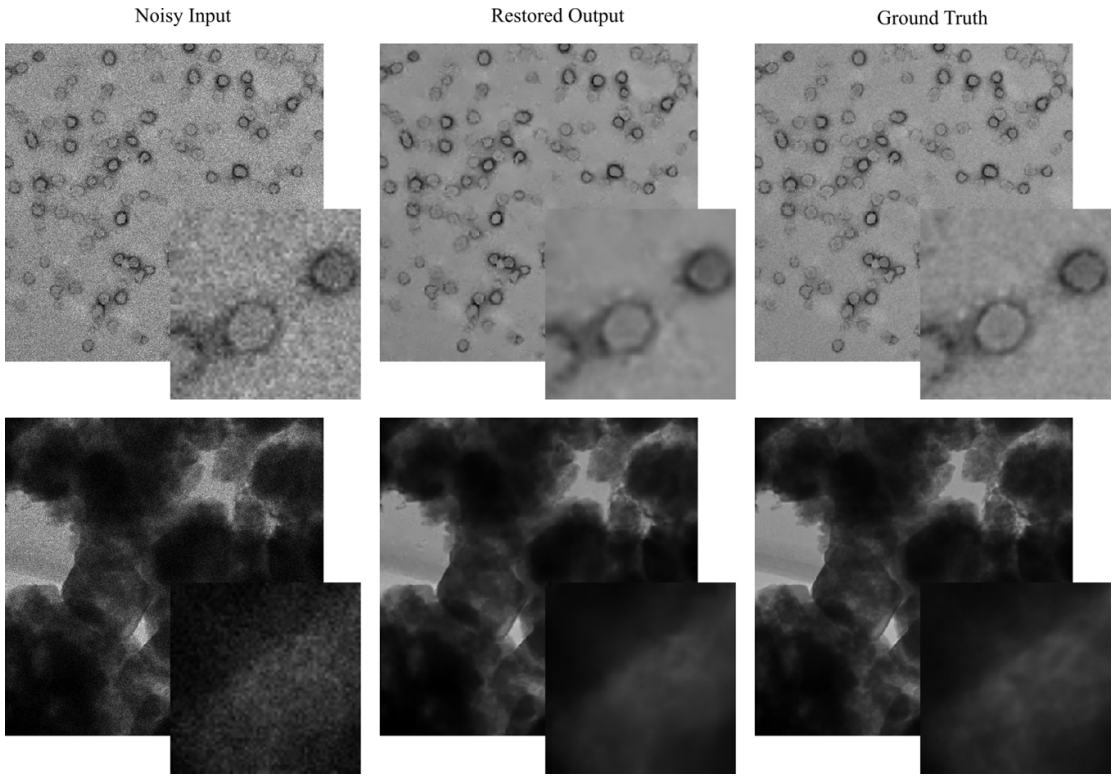

Figure 1. Example applications of a noise-removal DNN to instances of Poisson noise applied to 512 × 512 crops from TEM images. Enlarged 64 × 64 regions from the top left of each crop are shown to ease comparison. This figure is adapted from our earlier work [72] under a Creative Commons Attribution 4.0 [73] license.

programmatically describing a complicated reality. As a case in point, an ANN was able to outperform decades of advances in traditional denoising algorithms after training on two GPUs for a week [70].

Definitions of electron microscope noise can include statistical noise [128–135], aberrations [136], scan distortions [137–140], specimen drift [141], and electron beam damage [142]. Statistical noise is often minimized by either increasing electron dose or applying traditional denoising algorithms [143, 144]. There are a variety of denoising algorithms developed for electron microscopy, including algorithms based on block matching [145], contourlet transforms [114, 115], energy minimization [146], fast patch reorderings [147], Gaussian kernel density estimation [148], Kronecker envelope principal component analysis [149] (PCA), non-local means and Zernike moments [150], singular value thresholding [151], wavelets [152], and other approaches [141, 153–156]. Noise that is not statistical is often minimized by hardware. For example, by using aberration correctors [136, 157–159], choosing scanning transmission electron microscopy (STEM) scan shapes and speeds that minimize distortions [138], and using stable sample holders to reduce drift [160]. Beam damage can also be reduced by using minimal electron voltage and electron dose [161–163], or dose-fractionation across multiple frames in multi-pass transmission electron microscopy [164–166] (TEM) or STEM [167].

Deep learning is being applied to improve signal-to-noise for a variety of applications [168–176]. Most approaches in electron microscopy involve training ANNs to either map low-quality experimental [177], artificially deteriorated [70, 178] or synthetic [179–182] inputs to paired high-quality experimental measurements. For example, applications of a DNN trained with artificially deteriorated TEM images are shown in figure 1. However, ANNs have also been trained with unpaired datasets of low-quality and high-quality electron micrographs [183], or pairs of low-quality electron micrographs [184, 185]. Another approach is Noise2Void [168], ANNs are trained from single noisy images. However, Noise2Void removes information by masking noisy input pixels corresponding to target output pixels. So far, most ANNs that improve electron microscope signal-to-noise have been trained to decrease statistical noise [70, 177, 179–184, 186] as other approaches have been developed to correct electron microscope scan distortions [187, 188] and specimen drift [141, 188, 189]. However, we anticipate that ANNs will be developed to correct a variety of electron microscopy noise as ANNs have been developed for aberration correction of optical microscopy [190–195] and photoacoustic [196] signals.

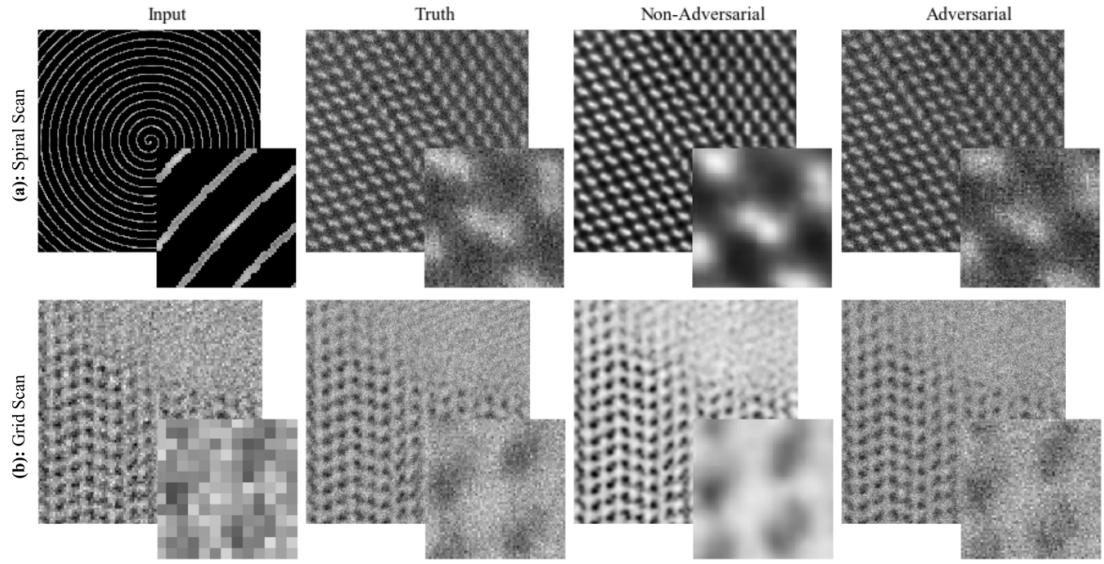

Figure 2. Example applications of DNNs to restore 512 × 512 STEM images from sparse signals. Training as part of a generative adversarial network [197–200] yields more realistic outputs than training a single DNN with mean squared errors. Enlarged 64 × 64 regions from the top left of each crop are shown to ease comparison. (a) Input is a Gaussian blurred 1/20 coverage spiral [201]. (b) Input is a 1/25 coverage grid [202]. This figure is adapted from our earlier works under Creative Commons Attribution 4.0 [73] licenses.

1.2. Compressed sensing

Compressed sensing [203–207] is the efficient reconstruction of a signal from a subset of measurements. Applications include faster medical imaging [208–210], image compression [211, 212], increasing image resolution [213, 214], lower medical radiation exposure [215–217], and low-light vision [218, 219]. In STEM, compressed sensing has enabled electron beam exposure and scan time to be decreased by 10–100× with minimal information loss [201, 202]. Thus, compressed sensing can be essential to investigations where the high current density of electron probes damages specimens [161, 220–226]. Even if the effects of beam damage can be corrected by postprocessing, the damage to specimens is often permanent. Examples of beam-sensitive materials include organic crystals [227], metal-organic frameworks [228], nanotubes [229], and nanoparticle dispersions [230]. In electron microscopy, compressed sensing is especially effective due to high signal redundancy [231]. For example, most electron microscopy images are sampled at 5–10× their Nyquist rates [232] to ease visual inspection, decrease sub-Nyquist aliasing [233], and avoid undersampling.

Perhaps the most popular approach to compressed sensing is upsampling or infilling a uniformly spaced grid of signals [234–236]. Interpolation methods include Lancsoz [234], nearest neighbour [237], polynomial interpolation [238], Wiener [239] and other resampling methods [240–242]. However, a variety of other strategies to minimize STEM beam damage have also been proposed, including dose fractionation [243] and a variety of sparse data collection methods [244]. Perhaps the most intensively investigated approach to the latter is sampling a random subset of pixels, followed by reconstruction using an inpainting algorithm [244–249]. Random sampling of pixels is nearly optimal for reconstruction by compressed sensing algorithms [250]. However, random sampling exceeds the design parameters of standard electron beam deflection systems, and can only be performed by collecting data slowly [138, 251], or with the addition of a fast deflection or blanking system [247, 252].

Sparse data collection methods that are more compatible with conventional STEM electron beam deflection systems have also been investigated. For example, maintaining a linear fast scan deflection whilst using a widely-spaced slow scan axis with some small random 'jitter' [245, 251]. However, even small jumps in electron beam position can lead to a significant difference between nominal and actual beam positions in a fast scan. Such jumps can be avoided by driving functions with continuous derivatives, such as those for spiral and Lissajous scan paths [138, 201, 247, 253, 254]. Sang *et al* [138, 254] considered a variety of scans including Archimedes and Fermat spirals, and scans with constant angular or linear displacements, by driving electron beam deflectors with a field-programmable gate array [255] (FPGA) based system [138]. Spirals with constant angular velocity place the least demand on electron beam deflectors. However, dwell times, and therefore electron dose, decreases with radius. Conversely, spirals created with constant spatial speeds are prone to systematic image distortions due to lags in deflector responses. In practice, fixed doses are preferable as they simplify visual inspection and limit the dose dependence of STEM noise [129].

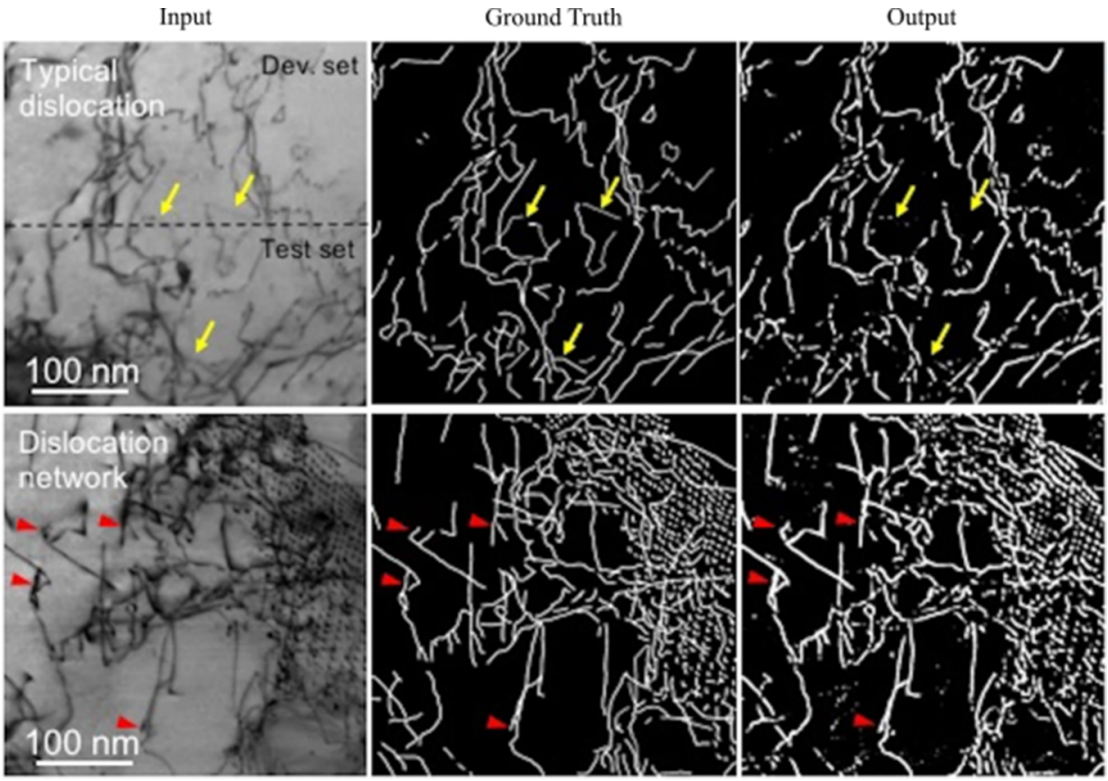

Figure 3. Example applications of a semantic segmentation DNN to STEM images of steel to classify dislocation locations. Yellow arrows mark uncommon dislocation lines with weak contrast, and red arrows indicate that fixed widths used for dislocation lines are sometimes too narrow to cover defects. This figure is adapted with permission [287] under a Creative Commons Attribution 4.0 [73] license.

Deep learning can leverage an understanding of physics to infill images [256–258]. Example applications include increasing scanning electron microscopy (SEM) [178, 259, 260], STEM [202, 261] and TEM [262] resolution, and infilling continuous sparse scans [201]. Example applications of DNNs to complete sparse spiral and grid scans are shown in figure 2. However, caution should be used when infilling large regions as ANNs may generate artefacts if a signal is unpredictable [201]. A popular alternative to deep learning for infilling large regions is exemplar-based infilling [263–266]. However, exemplar-based infilling often leaves artefacts [267] and is usually limited to leveraging information from single images. Smaller regions are often infilled by fast marching [268], Navier–Stokes infilling [269], or interpolation [238].

1.3. Labelling

Deep learning has been the basis of state-of-the-art classification [270–273] since convolutional neural networks (CNNs) enabled a breakthrough in classification accuracy on ImageNet [71]. Most classifiers are single feedforward neural networks (FNNs) that learn to predict discrete labels. In electron microscopy, applications include classifying image region quality [274, 275], material structures [276, 277], and image resolution [278]. However, siamese [279–281] and dynamically parameterized [282] networks can more quickly learn to recognise images. Finally, labelling ANNs can learn to predict continuous features, such as mechanical properties [283]. Labelling ANNs are often combined with other methods. For example, ANNs can be used to automatically identify particle locations [186, 284–286] to ease subsequent processing.

1.4. Semantic segmentation

Semantic segmentation is the classification of pixels into discrete categories. In electron microscopy, applications include the automatic identification of local features [288, 289], such as defects [290, 291], dopants [292], material phases [293], material structures [294, 295], dynamic surface phenomena [296], and chemical phases in nanoparticles [297]. Early approaches to semantic segmentation used simple rules. However, such methods were not robust to a high variety of data [298]. Subsequently, more adaptive algorithms based on soft-computing [299] and fuzzy algorithms [300] were developed to use geometric shapes as priors. However, these methods were limited by programmed features and struggled to handle the high variety of data.

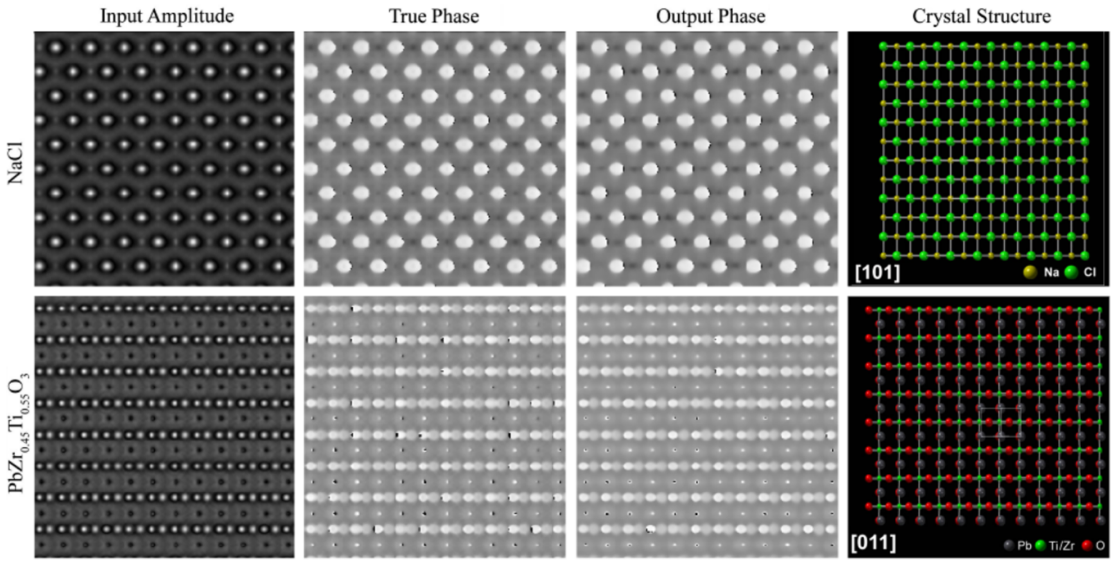

Figure 4. Example applications of a DNN to reconstruct phases of exit wavefunction from intensities of single TEM images. Phases in $[-\pi, \pi)$ rad are depicted on a linear greyscale from black to white, and Miller indices label projection directions. This figure is adapted from our earlier work [349] under a Creative Commons Attribution 4.0 [73] license.

To improve performance, DNNs have been trained to semantically segment images [301–308]. Semantic segmentation DNNs have been developed for focused ion beam scanning electron microscopy [309–311], SEM [311–314], STEM [287, 315], and TEM [286, 310, 311, 316–319]. For example, applications of a DNN to semantic segmentation of STEM images of steel are shown in figure 3. Deep learning based semantic segmentation also has a high variety of applications outside of electron microscopy, including autonomous driving [320–324], dietary monitoring [325, 326], magnetic resonance images [327–331], medical images [332–334] such as prenatal ultrasound [335–338], and satellite image translation [339–343]. Most DNNs for semantic segmentation are trained with images segmented by humans. However, human labelling may be too expensive, time-consuming, or inappropriate for sensitive data. Unsupervised semantic segmentation can avoid these difficulties by learning to segment images from an additional dataset of segmented images [344] or image-level labels [345–348]. However, unsupervised semantic segmentation networks are often less accurate than supervised networks.

1.5. Exit wavefunction reconstruction

Electrons exhibit wave-particle duality [350, 351], so electron propagation is often described by wave optics [352]. Applications of electron wavefunctions exiting materials [353] include determining projected potentials and corresponding crystal structure information [354, 355], information storage, point spread function deconvolution, improving contrast, aberration correction [356], thickness measurement [357], and electric and magnetic structure determination [358, 359]. Usually, exit wavefunctions are either iteratively reconstructed from focal series [360–364] or recorded by electron holography [352, 363, 365]. However, iterative reconstruction is often too slow for live applications, and holography is sensitive to distortions and may require expensive microscope modification.

Non-iterative methods based on DNNs have been developed to reconstruct optical exit wavefunctions from focal series [69] or single images [366–368]. Subsequently, DNNs have been developed to reconstruct exit wavefunctions from single TEM images [349], as shown in figure 4. Indeed, deep learning is increasingly being applied to accelerated quantum mechanics [369–374]. Other examples of DNNs adding new dimensions to data include semantic segmentation described in section 1.4, and reconstructing 3D atomic distortions from 2D images [375]. Non-iterative methods that do not use ANNs to recover phase information from single images have also been developed [376, 377]. However, they are limited to defocused images in the Fresnel regime [376], or to non-planar incident wavefunctions in the Fraunhofer regime [377].

2. Resources

Access to scientific resources is essential to scientific enterprise [378]. Fortunately, most resources needed to get started with machine learning are freely available. This section provides directions to various machine learning resources, including how to access deep learning frameworks (DLFs), a free GPU or tensor

Table 1. Deep learning frameworks with programming interfaces. Most frameworks have open source code and many support multiple programming languages.

Framework	License	Programming interfaces
Apache SINGA [420]	Apache 2.0 [421]	C++, Java, Python
BigDL [422]	Apache 2.0 [423]	Python, Scala
Caffe [424, 425]	BSD [426]	C++, MATLAB, Python
Chainer [427]	MIT [428]	Python
Deeplearning4j [429]	Apache 2.0 [430]	Clojure, Java, Kotlin, Python, Scala
Dlib [431, 432]	BSL [433]	C++
Flux [434]	MIT [435]	Julia
MATLAB Deep Learning Toolbox [436]	Proprietary [437]	MATLAB
Microsoft Cognitive Toolkit [438]	MIT [439]	BrainScript, C++, Python
Apache MXNet [440]	Apache 2.0 [441]	C++, Clojure, Go, JavaScript, Julia, Matlab, Perl, Python, R, Scala
OpenNN [442]	GNU LGPL [443]	C++
PaddlePaddle [444]	Apache 2.0 [445]	C++
PyTorch [446]	BSD [447]	C++, Python
TensorFlow [448, 449]	Apache 2.0 [450]	C++, C#, Go, Haskell, Julia, MATLAB, Python, Java, JavaScript, R, Ruby, Rust, Scala, Swift
Theano [451, 452]	BSD [453]	Python
Torch [454]	BSD [455]	C, Lua
Wolfram Mathematica [456]	Proprietary [457]	Wolfram Language

processing unit (TPU) to accelerate tensor computations, platforms that host datasets and source code, and pretrained models. To support the ideals of open science embodied by Plan S [378–380], we focus on resources that enhance collaboration and enable open access [381]. We also discuss how electron microscopes can interface with ANNs and the importance of machine learning resources in the context of electron microscopy. However, we expect that our insights into electron microscopy can be generalized to other scientific fields.

2.1. Hardware acceleration

A DNN is an ANN with multiple layers that perform a sequence of tensor operations. Tensors can either be computed on central processing units (CPUs) or hardware accelerators [62], such as FPGAs [382–385], GPUs [386–388], and TPUs [389–391]. Most benchmarks indicate that GPUs and TPUs outperform CPUs for typical DNNs that could be used for image processing [392–396] in electron microscopy. However, GPU and CPU performance can be comparable when CPU computation is optimized [397]. TPUs often outperform GPUs [394], and FPGAs can outperform GPUs [398, 399] if FPGAs have sufficient arithmetic units [400, 401]. Typical power consumption per TFLOPS [402] decreases in order CPU, GPU, FPGA, then TPU, so hardware acceleration can help to minimize long-term costs and environmental damage [403].

For beginners, Google Colab [404–407] and Kaggle [408] provide hardware accelerators in ready-to-go deep learning environments. Free compute time on these platforms is limited as they are not intended for industrial applications. Nevertheless, the free compute time is sufficient for some research [409]. For more intensive applications, it may be necessary to get permanent access to hardware accelerators. If so, many online guides detail how to install [410, 411] and set up an Nvidia [412] or AMD [413] GPU in a desktop computer for deep learning. However, most hardware comparisons for deep learning [414] focus on Nvidia GPUs as most DLFs use Nvidia's proprietary compute unified device architecture (CUDA) deep neural network (cuDNN) primitives for deep learning [415], which are optimized for Nvidia GPUs. Alternatively, hardware accelerators may be accessible from a university or other institutional high performance computing centre, or via a public cloud service provider [416–419].

2.2. Deep learning frameworks

A DLF [9, 458–464] is an interface, library or tool for DNN development. Features often include automatic differentiation [465], heterogeneous computing, pretrained models, and efficient computing [466] with CUDA [467–469], cuDNN [415, 470], OpenMP [471, 472], or similar libraries. Popular DLFs tabulated in table 1 often have open source code and support multiple programming interfaces. Overall, TensorFlow [448, 449] is the most popular DLF [473]. However, PyTorch [446] is the most popular DLF at top machine learning conferences [473, 474]. Some DLFs also have extensions that ease development or extend functionality. For example, TensorFlow extensions [475] that ease development include Keras [476], Sonnet

[477], Tensor2Tensor [478] and TFLearn [479, 480], and extensions that add functionality include Addons [481], Agents [482], Dopamine [483], Federated [484–486], Probability [487], and TRFL [488]. In addition, DLFs are supplemented by libraries for predictive data analysis, such as scikit-learn [489].

A limitation of the DLFs in table 1 is that users must use programming interfaces. This is problematic as many electron microscopists have limited, if any, programming experience. To increase accessibility, a range of graphical user interfaces (GUIs) have been created for ANN development. For example, ANNdotNET [490], Create ML [491], Deep Cognition [492], Deep Network Designer [493], DIGITS [494], ENNUI [495], Expresso [496], Neural Designer [497], Waikato environment for knowledge analysis [498–500] (WEKA) and ZeroCostDL4Mic [501]. The GUIs offer less functionality and scope for customization than programming interfaces. However, GUI-based DLFs are rapidly improving. Moreover, existing GUI functionality is more than sufficient to implement popular FNNs, such as image classifiers [272] and encoder–decoders [305–308, 502–504].

2.3. Pretrained models

Training ANNs is often time-consuming and computationally expensive [403]. Fortunately, pretrained models are available from a range of open access collections [505], such as Model Zoo [506], Open Neural Network Exchange [507–510] (ONNX) Model Zoo [511], TensorFlow Hub [512, 513], and TensorFlow Model Garden [514]. Some researchers also provide pretrained models via project repositories [70, 201, 202, 231, 349]. Pretrained models can be used immediately or to transfer learning [515–521] to new applications. For example, by fine-tuning and augmenting the final layer of a pretrained model [522]. Benefits of transfer learning can include decreasing training time by orders of magnitude, reducing training data requirements, and improving generalization [520, 523].

Using pretrained models is complicated by ANNs being developed with a variety of DLFs in a range of programming languages. However, most DLFs support interoperability. For example, by supporting the saving of models to a common format or to formats that are interoperable with the Neural Network Exchange Format [524] or ONNX formats. Many DLFs also support saving models to HDF5 [525, 526], which is popular in the pycroscopy [527, 528] and HyperSpy [529, 530] libraries used by electron microscopists. The main limitation of interoperability is that different DLFs may not support the same functionality. For example, Dlib [431, 432] does not support recurrent neural networks (RNNs) [531–536].

2.4. Datasets

Randomly initialized ANNs [537] must be trained, validated, and tested with large, carefully partitioned datasets to ensure that they are robust to general use [538]. Most ANN training starts from random initialization, rather than transfer learning [515–521], as follows.

(a) Researchers may be investigating modifications to ANN architecture or ability to learn.
(b) Pretrained models may be unavailable or too difficult to find.
(c) Models may quickly achieve sufficient performance from random initialization. For example, training an encoder-decoder based on Xception [539] to improve electron micrograph signal-to-noise [70] can require less training than for PASCAL VOC 2012 [540] semantic segmentation [305].
(d) There may be a high computing budget, so transfer learning is unnecessary [541, 542].

There are millions of open access datasets [543, 544] and a range of platforms that host [545–549] or aggregate [550–553] machine learning datasets. Openly archiving datasets drives scientific enterprise by reducing need to repeat experiments [554–558], enabling new applications through data mining [559, 560], and standardizing performance benchmarks [561]. For example, popular datasets used to standardize image classification performance benchmarks include CIFAR-10 [562, 563], MNIST [564] and ImageNet [565]. A high range of both domain-specific and general platforms that host scientific data for free are listed by the Open Access Directory [566] and Nature Scientific Data [567]. For beginners, we recommend Zenodo [568] as it is free, open access, has an easy-to-use interface, and will host an unlimited number of datasets smaller than 50 GB for at least 20 years [569].

There are a range of platforms dedicated to hosting electron microscopy datasets, including the Caltech Electron Tomography Database [570] (ETDB-Caltech), Electron Microscopy Data Bank [571–576] (EMDataBank), and the Electron Microscopy Public Image Archive [577] (EMPIAR). However, most electron microscopy datasets are small, esoteric or are not partitioned for machine learning [231]. Nevertheless, a variety of large machine learning datasets for electron microscopy are being published in independent repositories [231, 578, 579], including Warwick Electron Microscopy Datasets [231] that we curated. In addition, a variety of databases host information that supports electron microscopy. For example, crystal structure databases provide data in standard formats [580, 581], such as Crystallography Information

Table 2. Microjob service platforms. The size of typical tasks varies for different platforms and some platforms specialize in preparing machine learning datasets.

Platform	Website	For machine learning
Amazon Mechanical Turk	www.mturk.com	General tasks
Appen	https://appen.com	Machine learning data preparation
Clickworker	www.clickworker.com	Machine learning data preparation
Fiverr	www.fiverr.com	General tasks
Hive	https://thehive.ai	Machine learning data preparation
iMerit	https://imerit.net	Machine learning data preparation
JobBoy	www.jobboy.com	General tasks
Minijobz	https://minijobz.com	General tasks
Microworkers	www.microworkers.com	General tasks
OneSpace	https://freelance.onespace.com	General tasks
Playment	https://playment.io	Machine learning data preparation
RapidWorkers	https://rapidworkers.com	General tasks
Scale	https://scale.com	Machine learning data preparation
Smart Crowd	https://thesmartcrowd.lionbridge.com	General tasks
Trainingset.ai	www.trainingset.ai	Machine learning data preparation
ySense	www.ysense.com	General tasks

Files [582–585]. Large crystal structure databases [586–588] containing over 10^5 crystal structures include the Crystallography Open Database [589–594], Inorganic Crystal Structure Database [595–599], and National Institute of Standards and Technology Crystal Data [600, 601].

To achieve high performance, it may be necessary to curate a large dataset for ANN training [2]. However, large datasets like DeepMind Kinetics [602], ImageNet [565], and YouTube 8M [603] may take a team months to prepare. As a result, it may not be practical to divert sufficient staff and resources to curate a high-quality dataset, even if curation is partially automated [603–610]. To curate data, human capital can be temporarily and cheaply increased by using microjob services [611]. For example, through microjob platforms tabulated in table 2. Increasingly, platforms are emerging that specialize in data preparation for machine learning. Nevertheless, microjob services may be inappropriate for sensitive data or tasks that require substantial domain-specific knowledge.

2.5. Source code

Software is part of our cultural, industrial, and scientific heritage [612]. Source code should therefore be archived where possible. For example, on an open source code platform such as Apache Allura [613], AWS CodeCommit [614], Beanstalk [615], BitBucket [616], GitHub [617], GitLab [618], Gogs [619], Google Cloud Source Repositories [620], Launchpad [621], Phabricator [622], Savannah [623] or SourceForge [624]. These platforms enhance collaboration with functionality that helps users to watch [625] and contribute improvements [626–632] to source code. The choice of platform is often not immediately important for small electron microscopy projects as most platforms offer similar functionality. Nevertheless, functionality comparisons of open source platforms are available [633–635]. For beginners, we recommend GitHub as it is actively developed, scalable to large projects and has an easy-to-use interface.

2.6. Finding information

Most web traffic [636, 637] goes to large-scale web search engines [638–642] such as Bing, DuckDuckGo, Google, and Yahoo. This includes searches for scholarly content [643–645]. We recommend Google for electron microscopy queries as it appears to yield the best results for general [646–648], scholarly [644, 645] and other [649] queries. However, general search engines can be outperformed by dedicated search engines for specialized applications. For example, for finding academic literature [650–652], data [653], jobs [654, 655], publication venues [656], patents [657–660], people [661–663], and many other resources. The use of search engines is increasingly political [664–666] as they influence which information people see. However, most users appear to be satisfied with their performance [667].

Introductory textbooks are outdated [668, 669] insofar that most information is readily available online. We find that some websites are frequent references for up-to-date and practical information:

(a) Stack Overflow [670–675] is a source of working code snippets and a useful reference when debugging code.
(b) Papers With Code State-of-the-Art [561] leaderboards rank the highest performing ANNs with open source code for various benchmarks.

(c) Medium [676] and its subsidiaries publish blogs with up-to-date and practical advice about machine learning.
(d) The Machine Learning subreddit [677] hosts discussions about machine learning. In addition, there is a Learn Machine Learning subreddit [678] aimed at beginners.
(e) Dave Mitchell's DigitalMicrograph Scripting Website [679, 680] hosts a collection of scripts and documentation for programming electron microscopes.
(f) The Internet Archive [681, 682] maintains copies of software and media, including webpages via its Wayback Machine [683–685].
(g) Distill [686] is a journal dedicated to providing clear explanations about machine learning. Monetary prizes are awarded for excellent communication and refinement of ideas.

This list enumerates popular resources that we find useful, so it may introduce personal bias. However, alternative guides to useful resources are available [687–689]. We find that the most common issues finding information are part of an ongoing reproducibility crisis [690, 691] where machine learning researchers do not publish their source code or data. Nevertheless, third party source code is sometimes available. Alternatively, ANNs can reconstruct source code from some research papers [692].

2.7. Scientific publishing

The number of articles published per year in reputable peer-reviewed [693–697] scientific journals [698, 699] has roughly doubled every nine years since the beginning of modern science [700]. There are now over 25 000 peer-reviewed journals [699] with varying impact factors [701–703], scopes and editorial policies. Strategies to find the best journal to publish in include using online journal finders [704], seeking the advice of learned colleagues, and considering where similar research has been published. Increasingly, working papers are also being published in open access preprint archives [705–707]. For example, the arXiv [708, 709] is a popular preprint archive for computer science, mathematics, and physics. Advantages of preprints include ensuring that research is openly available, increasing discovery and citations [710–714], inviting timely scientific discussion, and raising awareness to reduce unnecessary duplication of research. Many publishers have adapted to the popularity of preprints [705] by offering open access publication options [715–718] and allowing, and in some cases encouraging [719], the prior publication of preprints. Indeed, some journals are now using the arXiv to host their publications [720].

A variety of software can help authors prepare scientific manuscripts [721]. However, we think the most essential software is a document preparation system. Most manuscripts are prepared with Microsoft Word [722] or similar software [723]. However, Latex [724–726] is a popular alternative among computer scientists, mathematicians and physicists [727]. Most electron microscopists at the University of Warwick appear to prefer Word. A 2014 comparison of Latex and Word found that Word is better at all tasks other than typesetting equations [728]. However, in 2017 it become possible to use Latex to typeset equations within Word [727]. As a result, Word appears to be more efficient than Latex for most manuscript preparation. Nevertheless, Latex may still be preferable to authors who want fine control over typesetting [729, 730]. As a compromise, we use Overleaf [731] to edit Latex source code, then copy our code to Word as part of proofreading to identify issues with grammar and wording.

3. Electron microscopy

An electron microscope is an instrument that uses electrons as a source of illumination to enable the study of small objects. Electron microscopy competes with a large range of alternative techniques for material analysis [732–734], including atomic force microscopy [735–737]; Fourier transformed infrared spectroscopy [738, 739]; nuclear magnetic resonance [740–743]; Raman spectroscopy [744–750]; and x-ray diffraction (XRD) [751, 752], dispersion [753], fluorescence [754, 755], and photoelectron spectroscopy [756, 757]. Quantitative advantages of electron microscopes can include higher resolution and depth of field, and lower radiation damage than light microscopes [758]. In addition, electron microscopes can record images, enabling visual interpretation of complex structures that may otherwise be intractable. This section will briefly introduce varieties of electron microscopes, simulation software, and how electron microscopes can interface with ANNs.

3.1. Microscopes

There are a variety of electron microscopes that use different illumination mechanisms. For example, reflection electron microscopy (REM) [759, 760], SEM [761, 762], STEM [763, 764], scanning tunnelling microscopy [765, 766] (STM), and TEM [767–769]. To roughly gauge popularities of electron microscope varieties, we performed abstract searches with Dimenions.ai [651, 770–772] for their abbreviations followed

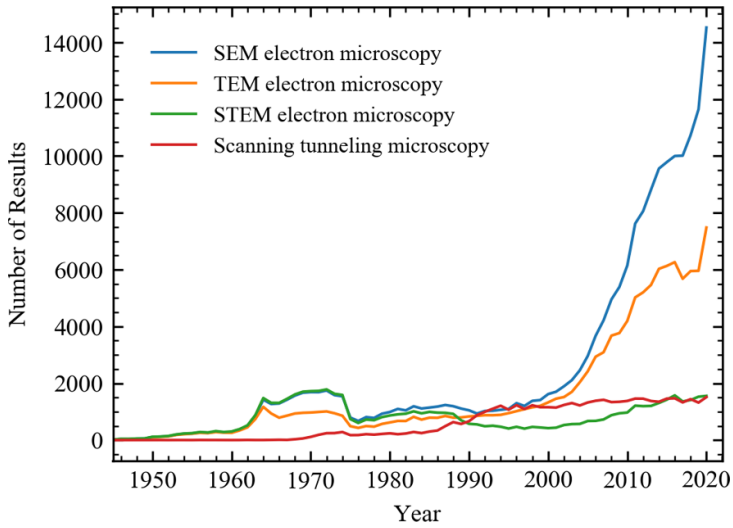

Figure 5. Numbers of results per year returned by Dimensions.ai abstract searches for SEM, TEM, STEM, STM and REM qualitate their popularities. The number of results for 2020 is extrapolated using the mean rate before 14 July 2020.

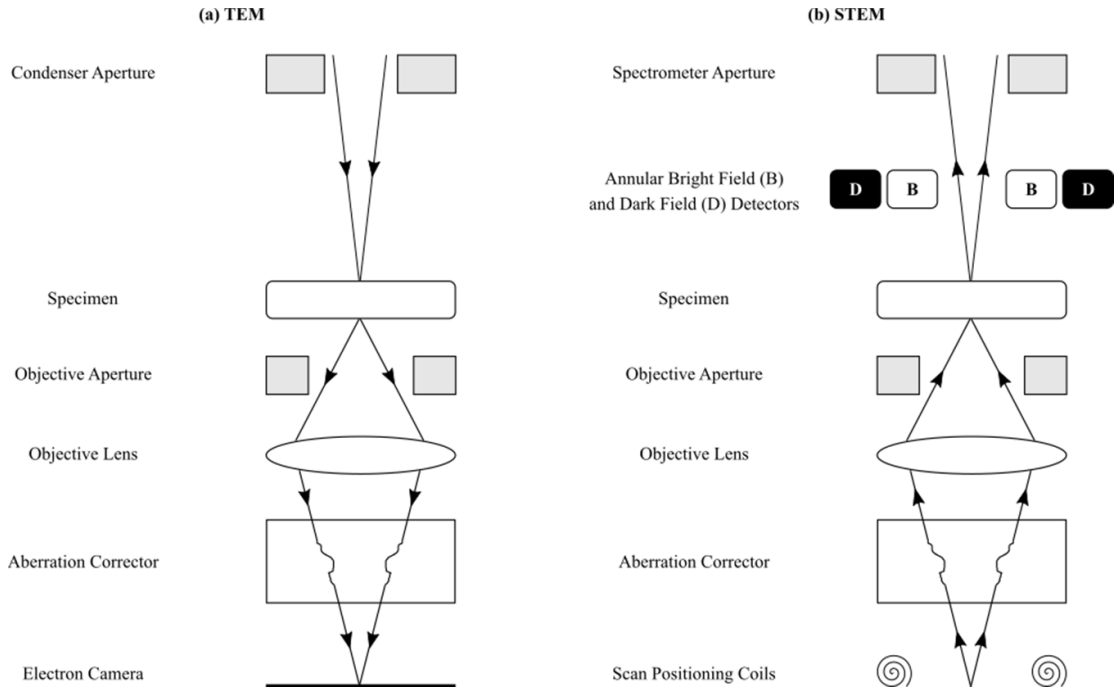

Figure 6. Reciprocity of TEM and STEM electron optics.

by 'electron microscopy' e.g. 'REM electron microscopy'. Numbers of results per year in figure 5 qualitate that popularity increases in order REM, STM, STEM, TEM, then SEM. It may be tempting to attribute the popularity of SEM over TEM to the lower cost of SEM [773], which increases accessibility. However, a range of considerations influence the procurement of electron microscopes [774] and hourly pricing at universities [775–779] is similar for SEM and TEM.

In SEM, material surfaces are scanned by sequential probing with a beam of electrons, which are typically accelerated to 0.2–40 keV. The SEM detects quanta emitted from where the beam interacts with the sample. Most SEM imaging uses low-energy secondary electrons. However, REM [759, 760] uses elastically backscattered electrons and is often complimented by a combination of reflection high-energy electron diffraction [780–782], reflection high-energy electron loss spectroscopy [783, 784] and spin-polarized low-energy electron microscopy [785–787]. Some SEMs also detect Auger electrons [788, 789]. To enhance materials characterization, most SEMs also detect light. The most common light detectors are for

cathodoluminescence and energy dispersive x-ray [790, 791] (EDX) spectroscopy. Nonetheless, some SEMs also detect Bremsstrahlung radiation [792].

Alternatively, TEM and STEM detect electrons transmitted through specimens. In conventional TEM, a single region is exposed to a broad electron beam. In contrast, STEM uses a fine electron beam to probe a series of discrete probing locations. Typically, electrons are accelerated across a potential difference to kinetic energies, E_k, of 80–300 keV. Electrons also have rest energy $E_e = m_e c^2$, where m_e is electron rest mass and c is the speed of light. The total energy, $E_t = E_e + E_k$, of free electrons is related to their rest mass energy by a Lorentz factor, γ,

$$E_t = \gamma m_e c^2 \,, \tag{1}$$

$$\gamma = (1 - v^2/c^2)^{1/2} \,, \tag{2}$$

where v is the speed of electron propagation in the rest frame of an electron microscope. Electron kinetic energies in TEM and STEM are comparable to their rest energy, $E_e = 511$ keV [793], so relativistic phenomena [794, 795] must be considered to accurately describe their dynamics.

Electrons exhibit wave-particle duality [350, 351]. Thus, in an ideal electron microscope, the maximum possible detection angle, θ, between two point sources separated by a distance, d, perpendicular to the electron propagation direction is diffraction-limited. The resolution limit for imaging can be quantified by Rayleigh's criterion [796–798]

$$\theta \simeq 1.22 \frac{\lambda}{d}, \tag{3}$$

where resolution increases with decreasing wavelength, λ. Electron wavelength increases with increasing accelerating voltage, as described by the relativistic de Broglie relation [799–801],

$$\lambda = hc \left(E_k^2 + 2E_e E_k\right)^{-1/2} \,, \tag{4}$$

where h is Planck's constant [793]. Electron wavelengths for typical acceleration voltages tabulated by JEOL are in picometres [802]. In comparison, Cu K-α x-rays, which are often used for XRD, have wavelengths near 0.15 nm [803]. In theory, electrons can therefore achieve over 100× higher resolution than x-rays. Electrons and x-rays are both ionizing; however, electrons often do less radiation damage to thin specimens than x-rays [758]. Tangentially, TEM and STEM often achieve over ten times higher resolution than SEM [804] as transmitted electrons in TEM and STEM are easier to resolve than electrons returned from material surfaces in SEM.

In practice, TEM and STEM are also limited by incoherence [805–807] introduced by inelastic scattering, electron energy spread, and other mechanisms. TEM and STEM are related by an extension of Helmholtz reciprocity [808, 809] where the source plane in a TEM corresponds to the detector plane in a STEM [810], as shown in figure 6. Consequently, TEM coherence is limited by electron optics between the specimen and image, whereas STEM coherence is limited by the illumination system. For conventional TEM and STEM imaging, electrons are normally incident on a specimen [811]. Advantages of STEM imaging can include higher contrast and resolution than TEM imaging, and lower radiation damage [812]. As a result, STEM is increasing being favoured over TEM for high-resolution studies. However, we caution that definitions of TEM and STEM resolution can be disparate [813].

In addition to conventional imaging, TEM and STEM include a variety of operating modes for different applications. For example, TEM operating configurations include electron diffraction [814]; convergent beam electron diffraction [815–817]; tomography [818–826]; and bright field [768, 827–829], dark field [768, 829] and annular dark field [830] imaging. Similarly, STEM operating configurations include differential phase contrast [831–834]; tomography [818, 820, 822, 823]; and bright field [835, 836] or dark field [837] imaging. Further, electron cameras [838, 839] are often supplemented by secondary signal detectors. For example, elemental composition is often mapped by EDX spectroscopy, electron energy loss spectroscopy [840, 841] or wavelength dispersive spectroscopy [842, 843]. Similarly, electron backscatter diffraction [844–846] can detect strain [847–849] and crystallization [850–852].

3.2. Contrast simulation

The propagation of electron wavefunctions though electron microscopes can be described by wave optics [136]. Following, the most popular approach to modelling measurement contrast is multislice simulation [853, 854], where an electron wavefunction is iteratively perturbed as it travels through a model of a

specimen. Multislice software for electron microscopy includes ACEM [854–856], clTEM [857, 858], cudaEM [859], Dr Probe [860, 861], EMSoft [862, 863], JEMS [864], JMULTIS [865], MULTEM [866–868], NCEMSS [869, 870], NUMIS [871], Prismatic [872–874], QSTEM [875], SimulaTEM [876], STEM-CELL [877], Tempas [878], and xHREM [879–884]. We find that most multislice software is a recreation and slight modification of common functionality, possibly due to a publish-or-perish culture in academia [885–887]. Bloch-wave simulation [854, 888–892] is an alternative to multislice simulation that can reduce computation time and memory requirements for crystalline materials [893].

3.3. Automation

Most modern electron microscopes support Gatan Microscopy Suite (GMS) Software [894]. GMS enables electron microscopes to be programmed by DigitalMicrograph Scripting, a propriety Gatan programming language akin to a simplified version of C++. A variety of DigitalMicrograph scripts, tutorials and related resources are available from Dave Mitchell's DigitalMicrograph Scripting Website [679, 680], FELMI/ZFE's Script Database [895] and Gatan's Script library [896]. Some electron microscopists also provide DigitalMicrograph scripting resources on their webpages [897–899]. However, DigitalMicrograph scripts are slow insofar that they are interpreted at runtime, and there is limited native functionality for parallel and distributed computing. As a result, extensions to DigitalMicrograph scripting are often developed in other programming languages that offer more functionality.

Historically, most extensions were developed in C++ [900]. This was problematic as there is limited documentation, the standard approach used outdated C++ software development kits such as Visual Studio 2008, and programming expertise required to create functions that interface with DigitalMicrograph scripts limited accessibility. To increase accessibility, recent versions of GMS now support python [901]. This is convenient as it enables ANNs developed with python to readily interface with electron microscopes. For ANNs developed with C++, users have the option to either create C++ bindings for DigitalMicrograph script or for python. Integrating ANNs developed in other programming languages is more complicated as DigitalMicrograph provides almost no support. However, that complexity can be avoided by exchanging files from DigitalMicrograph script to external libraries via a random access memory (RAM) disk [902] or secondary storage [903].

Increasing accessibility, there are collections of GMS plugins with GUIs for automation and analysis [897–899, 904]. In addition, various individual plugins are available [905–909]. Some plugins are open source, so they can be adapted to interface with ANNs. However, many high-quality plugins are proprietary and closed source, limiting their use to automation of data collection and processing. Plugins can also be supplemented by a variety of libraries and interfaces for electron microscopy signal processing. For example, popular general-purpose software includes ImageJ [910], pycroscopy [527, 528] and HyperSpy [529, 530]. In addition, there are directories for tens of general-purpose and specific electron microscopy programs [911–913].

4. Components

Most modern ANNs are configured from a variety of DLF components. To take advantage of hardware accelerators [62], most ANNs are implemented as sequences of parallelizable layers of tensor operations [914]. Layers are often parallelized across data and may be parallelized across other dimensions [915]. This section introduces popular non-linear activation functions, normalization layers, convolutional layers, and skip connections. To add insight, we provide comparative discussion and address some common causes of confusion.

4.1. Nonlinear activation

In general, DNNs need multiple layers to be universal approximators [37–45]. Nonlinear activation functions [916, 917] are therefore essential to DNNs as successive linear layers can be contracted to a single layer. Activation functions separate artificial neurons, similar to biological neurons [918]. To learn efficiently, most DNNs are tens or hundreds of layers deep [47, 919–921]. High depth increases representational capacity [47], which can help training by gradient descent as DNNs evolve as linear models [922] and nonlinearities can create suboptimal local minima where data cannot be fit by linear models [923]. There are infinitely many possible activation functions. However, most activation functions have low polynomial order, similar to physical Hamiltonians [47].

Most ANNs developed for electron microscopy are for image processing, where the most popular nonlinearities are rectifier linear units [924, 925] (ReLUs). The ReLU activation, $f(x)$, of an input, x, and its gradient, $\partial_x f(x)$, are

$$f(x) = \max(0, x) \tag{5a}$$

$$\frac{\partial f(x)}{\partial x} = \begin{cases} 0, & \text{if } x \leq 0 \\ 1, & \text{if } x > 0. \end{cases} \tag{5b}$$

Popular variants of ReLUs include Leaky ReLU [926],

$$f(x) = \max(\alpha x, x) \tag{6a}$$

$$\frac{\partial f(x)}{\partial x} = \begin{cases} \alpha, & \text{if } x \leq 0 \\ 1, & \text{if } x > 0 \end{cases} \tag{6b}$$

where α is a hyperparameter, parametric ReLU [22] (PreLU) where α is a learned parameter, dynamic ReLU where α is a learned function of inputs [927], and randomized leaky ReLU [928] where α is chosen randomly. Typically, learned PreLU α are higher the nearer a layer is to ANN inputs [22]. Motivated by limited comparisons that do not show a clear performance difference between ReLU and leaky ReLU [929], some blogs [930] argue against using leaky ReLU due to its higher computational requirements and complexity. However, an in-depth comparison found that leaky ReLU variants consistently slightly outperform ReLU [928]. In addition, the non-zero gradient of leaky ReLU for $x \leq 0$ prevents saturating, or 'dying', ReLU [931–933], where the zero gradient of ReLUs stops learning.

There are a variety of other piecewise linear ReLU variants that can improve performance. For example, ReLUh activations are limited to a threshold [934], h, so that

$$f(x) = \min(\max(0, x), h) \tag{7a}$$

$$\frac{\partial f(x)}{\partial x} = \begin{cases} 0, & \text{if } x \leq 0 \\ 1, & \text{if } 0 < x \leq h \\ 0, & \text{if } x > h. \end{cases} \tag{7b}$$

Thresholds near $h = 6$ are often effective, so popular choice is ReLU6. Another popular activation is concatenated ReLU [935], which is the concatenation of ReLU(x) and ReLU$(-x)$. Other ReLU variants include adaptive convolutional [936], bipolar [937], elastic [938], and Lipschitz [939] ReLUs. However, most ReLU variants are uncommon as they are more complicated than ReLU and offer small, inconsistent, or unclear performance gains. Moreover, it follows from the universal approximator theorems [37–45] that disparity between ReLU and its variants approaches zero as network depth increases.

In shallow networks, curved activation functions with non-zero Hessians often accelerate convergence and improve performance. A popular activation is the exponential linear unit [940],

$$f(x) = \begin{cases} \alpha(\exp(x) - 1), & \text{if } x \leq 0 \\ x, & \text{if } x \geq 0 \end{cases} \tag{8a}$$

$$\frac{\partial f(x)}{\partial x} = \begin{cases} \alpha \exp(x), & \text{if } x \leq 0 \\ 1, & \text{if } x \geq 0 \end{cases} \tag{8b}$$

where α is a learned parameter. Further, a scaled ELU (SELU) [941],

$$f(x) = \begin{cases} \lambda\alpha(\exp(x) - 1), & \text{if } x \leq 0 \\ \lambda x, & \text{if } x \geq 0 \end{cases} \tag{9a}$$

$$\frac{\partial f(x)}{\partial x} = \begin{cases} \lambda\alpha \exp(x), & \text{if } x \leq 0 \\ \lambda, & \text{if } x \geq 0 \end{cases} \tag{9b}$$

with fixed $\alpha = 1.67326$ and scale factor $\lambda = 1.0507$ can be used to create self-normalizing neural networks (SNNs). A SNN cannot be derived from ReLUs or most other activation functions. Activation functions with

curvature are especially common in ANNs with only a couple of layers. For example, activation functions in radial basis function (RBF) networks [942–945], which are efficient universal approximators, are often Gaussians, multiquadratics, inverse multiquadratics, or square-based RBFs [946]. Similarly, support vector machines [947–949] often use RBFs, or sigmoids,

$$f(x) = \frac{1}{1+\exp(-x)} \tag{10a}$$

$$\frac{\partial f(x)}{\partial x} = f(x)\,(1-f(x))\,. \tag{10b}$$

Sigmoids can also be applied to limit the support of outputs. Unscaled, or 'logistic', sigmoids are often denoted $\sigma(x)$ and are related to tanh by $\tanh(x) = 2\sigma(2x)-1$. To avoid expensive $\exp(-x)$ in the computation of tanh, we recommend K-tanH [950], LeCun tanh [951], or piecewise linear approximation [952, 953].

The activation functions introduced so far are scalar functions than can be efficiently computed in parallel for each input element. However, functions of vectors, $\mathbf{x} = \{x_1, x_2, \ldots\}$, are also popular. For example, softmax activation [954],

$$f(\mathbf{x}) = \frac{\exp(\mathbf{x})}{\mathrm{sum}(\exp(\mathbf{x}))} \tag{11a}$$

$$\frac{f(\mathbf{x})}{\partial x_j} = \sum_i f(\mathbf{x})_i(\delta_{ij} - f(\mathbf{x})_j) \tag{11b}$$

is often applied before computing cross-entropy losses for classification networks. Similarly, Ln vector normalization,

$$f(\mathbf{x}) = \frac{\mathbf{x}}{||\mathbf{x}||_n} \tag{12a}$$

$$\frac{f(\mathbf{x})}{\partial x_j} = \frac{1}{||\mathbf{x}||_n}\left(1 - \frac{x_j^n}{||\mathbf{x}||_n^n}\right) \tag{12b}$$

is often applied to n-dimensional vectors to ensure that they lie on a unit n-sphere [349]. Finally, max pooling [955, 956],

$$f(\mathbf{x}) = \max(\mathbf{x}) \tag{13a}$$

$$\frac{f(\mathbf{x})}{\partial x_j} = \begin{cases} 1, & \text{if } j = \mathrm{argmax}(\mathbf{x}) \\ 0, & \text{if } j \neq \mathrm{argmax}(\mathbf{x}) \end{cases} \tag{13b}$$

is another popular multivariate activation function that is often used for downsampling. However, max pooling has fallen out of favour as it is often outperformed by strided convolutional layers [957]. Other vector activation functions include squashing nonlinearities for dynamic routing by agreement in capsule networks [958] and cosine similarity [959].

There is a range of other activation functions that are not detailed here for brevity. Further, finding new activation functions is an active area of research [960, 961]. Notable variants include choosing activation functions from a set before training [962, 963] and learning activation functions [962, 964–967]. Activation functions can also encode probability distributions [968–970] or include noise [953]. Finally, there are a variety of other deterministic activation functions [961, 971]. In electron microscopy, most ANNs enable new or enhance existing applications. Subsequently, we recommend using computationally efficient and established activation functions unless there is a compelling reason to use a specialized activation function.

4.2. Normalization

Normalization [972–974] standardizes signals, which can accelerate convergence by gradient descent and improve performance. Batch normalization [975–980] is the most popular normalization layer in image processing DNNs trained with minibatches of N examples. Technically, a 'batch' is an entire training dataset and a 'minibatch' is a subset; however, the 'mini' is often omitted where meaning is clear from context. During training, batch normalization applies a transform,

$$\mu_B = \frac{1}{N}\sum_{i=1}^{N} x_i\,, \tag{14}$$

$$\sigma_B^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i-\mu_B)^2\,, \tag{15}$$

$$\hat{\mathbf{x}} = \frac{\mathbf{x}-\mu_B}{(\sigma_B^2+\epsilon)^{1/2}}\,, \tag{16}$$

$$\mathrm{BatchNorm}(\mathbf{x}) = \gamma\hat{\mathbf{x}}+\beta\,, \tag{17}$$

where $\mathbf{x} = \{x_1,\ldots,x_N\}$ is a batch of layer inputs, γ and β are a learnable scale and shift, and ε is a small constant added for numerical stability. During inference, batch normalization applies a transform,

$$\mathrm{BatchNorm}(\mathbf{x}) = \frac{\gamma}{(\mathrm{Var}[x]+\epsilon)^{1/2}}\mathbf{x}+\left(\beta-\frac{\gamma\mathrm{E}[x]}{(\mathrm{Var}[x]+\epsilon)^{1/2}}\right), \tag{18}$$

where E[x] and Var[x] are expected batch means and variances. For convenience, E[x] and Var[x] are often estimated with exponential moving averages that are tracked during training. However, E[x] and Var[x] can also be estimated by propagating examples through an ANN after training.

Increasing batch size stabilizes learning by averaging destabilizing loss spikes over batches [261]. Batched learning also enables more efficient utilization of modern hardware accelerators. For example, larger batch sizes improve utilization of GPU memory bandwidth and throughput [391, 981, 982]. Using large batches can also be more efficient than many small batches when distributing training across multiple CPU clusters or GPUs due to communication overheads. However, the performance benefits of large batch sizes can come at the cost of lower test accuracy as training with large batches tends to converge to sharper minima [983, 984]. As a result, it often best not to use batch sizes higher than $N \approx 32$ for image classification [985]. However, learning rate scaling [541] and layer-wise adaptive learning rates [986] can increase accuracy of training with fixed larger batch sizes. Batch size can also be increased throughout training without compromising accuracy [987] to exploit effective learning rates being inversely proportional to batch size [541, 987]. Alternatively, accuracy can be improved by creating larger batches from replicated instances of training inputs with different data augmentations [988].

There are a few caveats to batch normalization. Originally, batch normalization was applied before activation [976]. However, applying batch normalization after activation often slightly improves performance [989, 990]. In addition, training can be sensitive to the often-forgotten ε hyperparameter [991] in equation (16). Typically, performance decreases as ε is increased above $\varepsilon \approx 0.001$; however, there is a sharp increase in performance around $\varepsilon = 0.01$ on ImageNet. Finally, it is often assumed that batches are representative of the training dataset. This is often approximated by shuffling training data to sample independent and identically distributed (i.i.d.) samples. However, performance can often be improved by prioritizing sampling [992, 993]. We observe that batch normalization is usually effective if batch moments, μ_B and σ_B, have similar values for every batch.

Batch normalization is less effective when training batch sizes are small, or do not consist of independent samples. To improve performance, standard moments in equation (16) can be renormalized [994] to expected means, μ, and standard deviations, σ,

$$\hat{\mathbf{x}} \leftarrow r\hat{\mathbf{x}}+d\,, \tag{19}$$

$$r = \mathrm{clip}_{[1/r_{\max},r_{\max}]}\left(\frac{\sigma_B}{\sigma}\right), \tag{20}$$

$$d = \mathrm{clip}_{[-d_{\max},d_{\max}]}\left(\frac{\mu_B-\mu}{\sigma}\right), \tag{21}$$

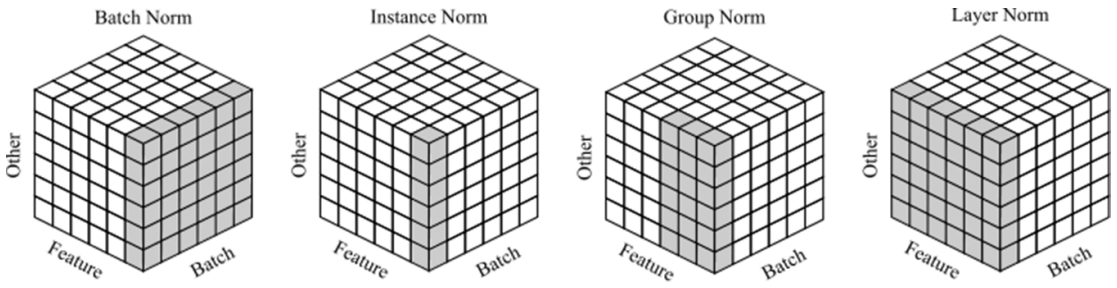

Figure 7. Visual comparison of various normalization methods highlighting regions that they normalize. Regions can be normalized across batch, feature and other dimensions, such as height and width.

where gradients are not backpropagated with respect to (w.r.t.) the renormalization parameters, r and d. Moments, μ and σ are tracked by exponential moving averages and clipping to $r_{\max}$ and $d_{\max}$ improves learning stability. Usually, clipping values are increased from starting values of $r_{\max} = 1$ and $d_{\max} = 0$, which correspond to batch normalization, as training progresses. Another approach is virtual batch normalization [995] (VBN), which estimates μ and σ from a reference batch of samples and does not require clipping. However, VBN is computationally expensive as it requires computing a second batch of statistics at every training iteration. Finally, online [996] and streaming [974] normalization enable training with small batch sizes by replace μ_B and σ_B in equation (16) with their exponential moving averages.

There are alternatives to the L_2 batch normalization of equations (14)–(18) that standardize to different Euclidean norms. For example, L_1 batch normalization [997] computes

$$s_1 = \frac{1}{N}\sum_{i=1}^{N} |x_i - \mu_B| \,, \tag{22}$$

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \mu_B}{C_{L_1} s_1} \,, \tag{23}$$

where $C_{L_1} = (\pi/2)^{1/2}$. Although the C_{L_1} factor could be learned by ANN parameters, its inclusion accelerates convergence of the original implementation of L_1 batch normalization [997]. Another alternative is L_∞ batch normalization [997], which computes

$$s_\infty = \text{mean}(\text{top}_k(|\mathbf{x} - \mu_B|)) \,, \tag{24}$$

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \mu_B}{C_{L_\infty} s_\infty} \,, \tag{25}$$

where C_{L_∞} is a scale factor, and $\text{top}_k(\mathbf{x})$ returns the k highest elements of $\mathbf{x}$. Hoffer *et al* suggest $k = 10$ [997]. Some L_1 batch normalization proponents claim that L_1 batch normalization outperforms [975] or achieves similar performance [997] to L_2 batch normalization. However, we found that L_1 batch normalization often lowers performance in our experiments. Similarly, L_∞ batch normalization often lowers performance [997]. Overall, L_1 and L_∞ batch normalization do not appear to offer a substantial advantage over L_2 batch normalization.

A variety of layers normalize samples independently, including layer, instance, and group normalization. They are compared with batch normalization in figure 7. Layer normalization [998, 999] is a transposition of batch normalization that is computed across feature channels for each training example, instead of across batches. Batch normalization is ineffective in RNNs; however, layer normalization of input activations often improves accuracy [998]. Instance normalization [1000] is an extreme version of layer normalization that standardizes each feature channel for each training example. Instance normalization was developed for style transfer [1001–1005] and makes ANNs insensitive to input image contrast. Group normalization [1006] is intermediate to instance and layer normalization insofar that it standardizes groups of channels for each training example.

The advantages of a set of multiple different normalization layers, Ω, can be combined by switchable normalization [1007, 1008], which standardizes to

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \sum_{z\in\Omega} \lambda_z^\mu \mu_z}{\sum_{z\in\Omega} \lambda_z^\sigma \sigma_z} \,, \tag{26}$$

where μ_z and σ_z are means and standard deviations computed by normalization layer z, and their respective importance ratios, λ_z^{μ} and λ_z^{σ}, are trainable parameters that are softmax activated to sum to unity. Combining batch and instance normalization statistics outperforms batch normalization for a range of computer vision tasks [1009]. However, most layers strongly weighted either batch or instance normalization, with most preferring batch normalization. Interestingly, combining batch, instance and layer normalization statistics [1007, 1008] results in instance normalization being preferred in earlier layers, whereas layer normalization was preferred in the later layers, and batch normalization was preferred in the middle layers. Smaller batch sizes lead to a preference towards layer normalization and instance normalization. Limitingly, using multiple normalization layers increases computation. To limit expense, we therefore recommend either defaulting to batch normalization, or progressively using single instance, batch or layer normalization layers.

A significant limitation of batch normalization is that it is not effective in RNNs. This is a limited issue as most electron microscopists are developing CNNs for image processing. However, we anticipate that RNNs may become more popular in electron microscopy following the increasing popularity of reinforcement learning (RL) [1010]. In addition to general-purpose alternatives to batch normalization that are effective in RNNs, such as layer normalization, there are a variety of dedicated normalization schemes. For example, recurrent batch normalization [1011, 1012] uses distinct normalization layers for each time step. Alternatively, batch normalized RNNs [1013] only have normalization layers between their input and hidden states. Finally, online [996] and streaming [974] normalization are general-purpose solutions that improve the performance of batch normalization in RNNs by applying batch normalization based on a stream of past batch statistics.

Normalization can also standardize trainable weights, $\mathbf{w}$. For example, weight normalization [1014],

$$\text{WeightNorm}(\mathbf{w}) = \frac{g}{||\mathbf{w}||_2}\mathbf{w}, \tag{27}$$

decouples the L2 norm, g, of a variable from its direction. Similarly, weight standardization [1015] subtracts means from variables and divides them by their standard deviations,

$$\text{WeightStd}(\mathbf{w}) = \frac{\mathbf{w} - \text{mean}(\mathbf{w})}{\text{std}(\mathbf{w})}, \tag{28}$$

similar to batch normalization. Weight normalization often outperforms batch normalization at small batch sizes. However, batch normalization consistently outperforms weight normalization at larger batch sizes used in practice [1016]. Combining weight normalization with running mean-only batch normalization can accelerate convergence [1014]. However, similar final accuracy can be achieved without mean-only batch normalization at the cost of slower convergence, or with the use of zero-mean preserving activation functions [937, 997]. To achieve similar performance to batch normalization, norm-bounded weight normalization [997] can be applied to DNNs with scale-invariant activation functions, such as ReLU. Norm-bounded weight normalization fixes g at initialization to avoid learning instability [997, 1016], and scales outputs with the final DNN layer.

Limitedly, weight normalization encourages the use of a small number of features to inform activations [1017]. To maximize feature utilization, spectral normalization [1017],

$$\text{SpectralNorm}(\mathbf{w}) = \frac{\mathbf{w}}{\sigma(\mathbf{w})}, \tag{29}$$

divides tensors by their spectral norms, $\sigma(\mathbf{w})$. Further, spectral normalization limits Lipschitz constants [1018], which often improves generative adversarial network [197–200] (GAN) training by bounding backpropagated discriminator gradients [1017]. The spectral norm of $\mathbf{v}$ is the maximum value of a diagonal matrix, $\mathbf{\Sigma}$, in the singular value decomposition [1019–1022],

$$\mathbf{v} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*, \tag{30}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices of orthonormal eigenvectors for $\mathbf{v}\mathbf{v}^T$ and $\mathbf{v}^T\mathbf{v}$, respectively. To minimize computation, $\boldsymbol{\sigma}(\mathbf{w})$ is often approximated by the power iteration method [1023, 1024],

$$\hat{\mathbf{v}} \leftarrow \frac{\mathbf{w}^{\mathrm{T}}\hat{\mathbf{u}}}{||\mathbf{w}^{\mathrm{T}}\hat{\mathbf{u}}||_2}, \tag{31}$$

$$\hat{\mathbf{u}} \leftarrow \frac{\mathbf{w}\hat{\mathbf{v}}}{||\mathbf{w}\hat{\mathbf{v}}||_2}, \tag{32}$$

$$\sigma(\mathbf{w}) \simeq \hat{\mathbf{u}}^T \mathbf{w} \hat{\mathbf{v}}, \tag{33}$$

where one iteration of equations (31) and (32) per training iteration is usually sufficient.

Parameter normalization can complement or be combined with signal normalization. For example, scale normalization [1025],

$$\text{ScaleNorm}(\mathbf{x}) = \frac{g}{||\mathbf{x}||_2}\mathbf{x}, \tag{34}$$

learns scales, g, for activations, and is often combined with weight normalization [1014, 1026] in transformer networks. Similarly, cosine normalization [959],

$$\text{CosineNorm}(\mathbf{x}) = \frac{\mathbf{w}}{||\mathbf{w}||_2} \cdot \frac{\mathbf{x}}{||\mathbf{x}||_2}, \tag{35}$$

computes products of L2 normalized parameters and signals. Both scale and cosine normalization can outperform batch normalization.

4.3. Convolutional layers

A CNN [1027–1030] is trained to weight convolutional kernels to exploit local correlations, such as spatial correlations in electron micrographs [231]. Historically, the development of CNNs was inspired by primate visual cortices [1031], where partially overlapping neurons are only stimulated by visual stimuli within their receptive fields. Based on this idea, Fukushima published his Neocognitron [1032–1035] in 1980. Convolutional formulations were then published by Atlas *et al* in 1988 for a single-layer CNN [1036], and LeCun *et al* in 1998 for a multi-layer CNN [1037, 1038]. Following, GPUs were applied to accelerate convolutions in 2010 [1039], leading to a breakthrough in classification performance on ImageNet with AlexNet in 2012 [71]. Indeed, the deep learning era is often partitioned into before and after AlexNet [19]. Deep CNNs are now ubiquitous. For example, there are review papers on applications of CNNs to action recognition in videos [1040], cytometry [1041], image and video compression [1042, 1043], image background subtraction [1044], image classification [272], image style transfer [1001], medical image analysis [332–334, 1045–1052], object detection [1053, 1054], semantic image segmentation [304, 332–334], and text classification [1055].

In general, the convolution of two functions, f and g, is

$$(f * g)(x) := \int_{s \in \Omega} f(s) g(x - s)\, \mathrm{d}s, \tag{36}$$

and their cross-correlation is

$$(f \circ g)(x) := \int_{s \in \Omega} f(s) g(x + s)\, \mathrm{d}s, \tag{37}$$

where integrals have unlimited support, Ω. In a CNN, convolutional layers sum convolutions of feature channels with trainable kernels, as shown in figure 8. Thus, f and g are discrete functions and the integrals in equations (36) and (37) can be replaced with limited summations. Since cross-correlation is equivalent to convolution if the kernel is flipped in every dimension, and CNN kernels are usually trainable, convolution and cross-correlation is often interchangeable in deep learning. For example, a TensorFlow function named 'tf.nn.convolution' computes cross-correlations [1056]. Nevertheless, the difference between convolution and cross-correlation can be source of subtle errors if convolutional layers from a DLF are used in an image processing pipeline with static asymmetric kernels.

Kernels designed by humans [1057] are often convolved in image processing pipelines. For example, convolutions of electron micrographs with Gaussian and Sobel kernels are shown in figure 9. Gaussian kernels compute local averages, blurring images and suppressing high-frequency noise. For example, a 5×5 symmetric Gaussian kernel with a 2.5 px standard deviation is

$$\begin{bmatrix} 0.1689 \\ 0.2148 \\ 0.2326 \\ 0.2148 \\ 0.1689 \end{bmatrix} \begin{bmatrix} 0.1689 & 0.2148 & 0.2326 & 0.2148 & 0.1689 \end{bmatrix} = \begin{bmatrix} 0.0285 & 0.0363 & 0.0393 & 0.0363 & 0.0285 \\ 0.0363 & 0.0461 & 0.0500 & 0.0461 & 0.0363 \\ 0.0393 & 0.0500 & 0.0541 & 0.0500 & 0.0393 \\ 0.0363 & 0.0461 & 0.0500 & 0.0461 & 0.0363 \\ 0.0285 & 0.0363 & 0.0393 & 0.0363 & 0.0285 \end{bmatrix}. \tag{38}$$

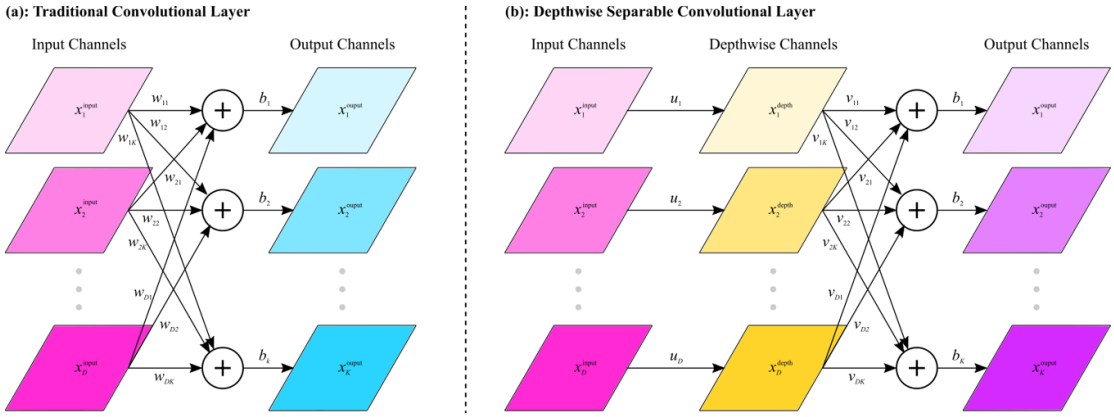

Figure 8. Visualization of convolutional layers. (a) Traditional convolutional layer where output channels are sums of biases and convolutions of weights with input channels. (b) Depthwise separable convolutional layer where depthwise convolutions compute one convolution with weights for each input channel. Output channels are sums of biases and pointwise convolutions weights with depthwise channels.

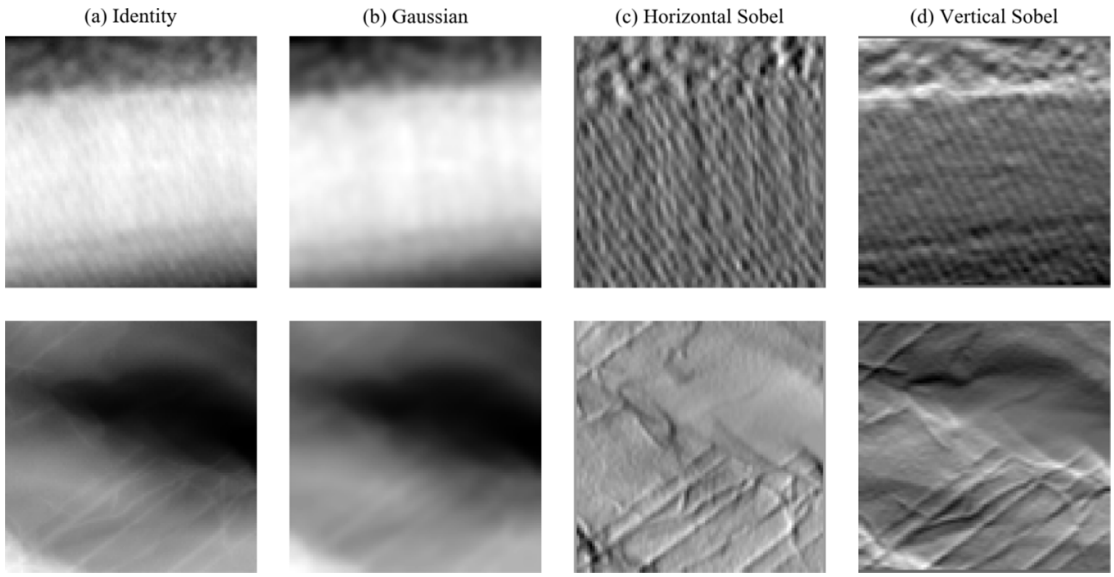

Figure 9. Two 96 × 96 electron micrographs (a) unchanged, and filtered by (b) a 5 × 5 symmetric Gaussian kernel with a 2.5 px standard deviation, (c) a 3 × 3 horizontal Sobel kernel, and (d) a 3 × 3 vertical Sobel kernel. Intensities in (a) and (b) are in [0, 1], whereas intensities in (c) and (d) are in [−1, 1].

Alternatives to Gaussian kernels for image smoothing [1058] include mean, median and bilateral filters. Sobel kernels compute horizontal and vertical spatial gradients that can be used for edge detection [1059]. For example, 3 × 3 Sobel kernels are

$$\begin{bmatrix}1\\2\\1\end{bmatrix}\begin{bmatrix}1 & 0 & -1\end{bmatrix} = \begin{bmatrix}1 & 0 & -1\\2 & 0 & -2\\1 & 0 & -1\end{bmatrix} \tag{39a}$$

$$\begin{bmatrix}1\\0\\-1\end{bmatrix}\begin{bmatrix}1 & 2 & 1\end{bmatrix} = \begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}. \tag{39b}$$

Alternatives to Sobel kernels offer similar utility, and include extended Sobel [1060], Scharr [1061, 1062], Kayyali [1063], Roberts cross [1064] and Prewitt [1065] kernels. Two-dimensional Gaussian and Sobel kernels are examples of linearly separable, or 'flattenable', kernels, which can be split into two one-dimensional kernels, as shown in equations (38)–(39*b*). Kernel separation can decrease computation in convolutional layers by convolving separated kernels in series, and CNNs that only use separable convolutions are effective [1066–1068]. However, serial convolutions decrease parallelization and separable kernels have

fewer degrees of freedom, decreasing representational capacity. Following, separated kernels are usually at least 5×5, and separated 3×3 kernels are unusual. Even-sized kernels, such as 2×2 and 4×4, are rare as symmetric padding is needed to avoid information erosion caused by spatial shifts of feature maps [1069].

A traditional 2D convolutional layer maps inputs, x^{input}, with height H, width, W, and depth, D, to

$$x_{kij}^{\text{output}} = b_k + \sum_{d=1}^{D} \sum_{m=1}^{M} \sum_{n=1}^{N} w_{dkmn} x_{d(i+m-1)(j+n-1)}^{\text{input}}, i \in [1, H-M+1], j \in [1, W-N+1], \tag{40}$$

where K output channels are indexed by $k \in [1, K]$, is the sum of a bias, b, and convolutions of each input channel with $M \times N$ kernels with weights, w. For clarity, a traditional convolutional layer is visualized in figure 8(a). Convolutional layers for 1D, 3D and higher-dimensional kernels [1070] have a similar form to 2D kernels, where kernels are convolved across each dimension. Most inputs to convolutional layers are padded [1071, 1072] to avoid reducing spatial resolutions by kernel sizes, which could remove all resolution in deep networks. Padding is computationally inexpensive and eases implementations of ANNs that would otherwise combine layers with different sizes, such as FractalNet [1073], Inception [1074–1076], NASNet [1077], recursive CNNs [1078, 1079], and ResNet [1080]. Pre-padding inputs results in higher performance than post-padding outputs [1081]. Following AlexNet [71], most convolutional layers are padded with zeros for simplicity. Reflection and replication padding achieve similar results to zero padding [1072]. However, padding based on partial convolutions [1082] consistently outperforms other methods [1072].

Convolutional layers are similar to fully connected layers used in multilayer perceptrons [1083, 1084] (MLPs). For comparison with equation (40), a fully connected, or 'dense', layer in a MLP computes

$$x_k^{\text{output}} = b_k + \sum_{d=1}^{D} w_{dk} x_d^{\text{input}}, \tag{41}$$

where every input element is connected to every output element. Convolutional layers reduce computation by making local connections within receptive fields of convolutional kernels, and by convolving kernels rather than using different weights at each input position. Intermediately, fully connected layers can be regularized to learn local connections [1085]. Fully connected layers are sometimes used at the middle of encoder-decoders [1086]. However, such fully connected layers can often be replaced by multiscale atrous, or 'holey', convolutions [955] in an atrous spatial pyramid pooling [305, 306] module to decrease computation without a significant decrease in performance. Alternatively, weights in fully connected layers can be decomposed into multiple smaller tensors to decrease computation without significantly decreasing performance [1087, 1088].

Convolutional layers can perform a variety of convolutional arithmetic [955]. For example, strided convolutions [1089] usually skip computation of outputs that are not at multiples of an integer spatial stride. Most strided convolutional layers are applied throughout CNNs to sequentially decrease spatial extent, and thereby decrease computational requirements. In addition, strided convolutions are often applied at the start of CNNs [539, 1074–1076] where most input features can be resolved at a lower resolution than the input. For simplicity and computational efficiency, stride is typically constant within a convolutional layer; however, increasing stride away from the centre of layers can improve performance [1090]. To increase spatial resolution, convolutional layers often use reciprocals of integer strides [1091]. Alternatively, spatial resolution can be increased by combining interpolative upsampling with an unstrided convolutional layer [1092, 1093], which can help to minimize output artefacts.

Convolutional layers couple the computation of spatial and cross-channel convolutions. However, partial decoupling of spatial and cross-channel convolutions by distributing inputs across multiple convolutional layers and combining outputs can improve performance. Partial decoupling of convolutions is prevalent in many seminal DNN architectures, including FractalNet [1073], Inception [1074–1076], NASNet [1077]. Taking decoupling to an extreme, depthwise separable convolutions [539, 1094, 1095] shown in figure 8 compute depthwise convolutions,

$$x_{dij}^{\text{depth}} = \sum_{m=1}^{M} \sum_{n=1}^{N} u_{dmn} x_{d(i+m-1)(j+n-1)}^{\text{input}}, i \in [1, H-M+1], j \in [1, W-N+1], \tag{42}$$

then compute pointwise 1×1 convolutions for D intermediate channels,

$$x_{kij}^{\text{output}} = b_k + \sum_{d=1}^{D} v_{dk}^{\text{point}} x_{dij}^{\text{depth}}, \tag{43}$$

where K output channels are indexed by $k \in [1, K]$. Depthwise convolution kernels have weights, u, and the depthwise layer is often followed by extra batch normalization before pointwise convolution to improve performance and accelerate convergence [1094]. Increasing numbers of channels with pointwise convolutions can increase accuracy [1094], at the cost of increased computation. Pointwise convolutions are a special case of traditional convolutional layers in equation (40) and have convolution kernel weights, v, and add biases, b. Naively, depthwise separable convolutions require fewer weight multiplications than traditional convolutions [1096, 1097]. However, extra batch normalization and serialization of one convolutional layer into depthwise and pointwise convolutional layers mean that depthwise separable convolutions and traditional convolutions have similar computing times [539, 1097].

Most DNNs developed for computer vision use fixed-size inputs. Although fixed input sizes are often regarded as an artificial constraint, it is similar to animalian vision where there is an effectively constant number of retinal rods and cones [1098–1100]. Typically, the most practical approach to handle arbitrary image shapes is to train a DNN with crops so that it can be tiled across images. In some cases, a combination of cropping, padding and interpolative resizing can also be used. To fully utilize unmodified variable size inputs, a simple is approach to train convolutional layers on variable size inputs. A pooling layer, such as global average pooling, can then be applied to fix output size before fully connected or other layers that might require fixed-size inputs. More involved approaches include spatial pyramid pooling [1101] or scale RNNs [1102]. Typical electron micrographs are much larger than 300×300, which often makes it unfeasible for electron microscopists with a few GPUs to train high-performance DNNs on full-size images. For comparison, Xception was trained on 300×300 images with 60 K80 GPUs for over one month.

The Fourier transform [1103], $\hat{f}(k_1, \ldots, k_N)$, at an N-dimensional Fourier space vector, $\{k_1, \ldots, k_N\}$, is related to a function, $f(x_1, \ldots, x_N)$, of an N-dimensional signal domain vector, $\{x_1, \ldots, x_N\}$, by

$$\hat{f}(k_1, \ldots, k_N) = \left(\frac{|b|}{(2\pi)^{1-a}}\right)^{N/2} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(x_1, \ldots, x_N) \exp(+ibk_1x_i + \cdots + ibk_Nx_N)\, \mathrm{d}x_1 \ldots \mathrm{d}x_N, \tag{44}$$

$$f(x_1, \ldots, x_N) = \left(\frac{|b|}{(2\pi)^{1+a}}\right)^{N/2} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \hat{f}(k_1, \ldots, k_N) \exp(-ibk_1x_i - \cdots - ibk_Nx_N)\, \mathrm{d}k_1 \ldots \mathrm{d}k_N, \tag{45}$$

where $\pi = 3.141\ldots$, and $i = (-1)^{1/2}$ is the imaginary number. Two parameters, a and b, can parameterize popular conventions that relate the Fourier and inverse Fourier transforms. Mathematica documentation nominates conventions [1104] for general applications (a, b), pure mathematics $(1,-1)$, classical physics $(-1, 1)$, modern physics $(0, 1)$, systems engineering $(1,-1)$, and signal processing $(0, 2\pi)$. We observe that most electron microscopists follow the modern physics convention of $a = 0$ and $b = 1$; however, the choice of convention is arbitrary and does not matter if it is consistent within a project. For discrete functions, Fourier integrals are replaced with summations that are limited to the support of a function.

Discrete Fourier transforms of uniformly spaced inputs are often computed with a FFT algorithm, which can be parallelized for CPUs [1105] or GPUs [65, 1106–1108]. Typically, the speedup of FFTs on GPUs over CPUs is higher for larger signals [1109, 1110]. Most popular FFTs are based on the Cooley–Turkey algorithm [1111, 1112], which recursively divides FFTs into smaller FFTs. We observe that some electron microscopists consider FFTs to be limited to radix-2 signals that can be recursively halved; however, FFTs can use any combination of factors for the sizes of recursively smaller FFTs. For example, clFFT [1113] FFT algorithms support signal sizes that are any sum of powers of 2, 3, 5, 7, 11 and 13.

Convolution theorems can decrease computation by enabling convolution in the Fourier domain [1114]. To ease notation, we denote the Fourier transform of a signal, $\mathbf{I}$, by $\mathrm{FT}(\mathbf{I})$, and the inverse Fourier transform by $\mathrm{FT}^{-1}(\mathbf{I})$. Following, the convolution theorems for two signals, $\mathbf{I}_1$ and $\mathbf{I}_2$, are [1115]

$$\mathrm{FT}(\mathbf{I}_1 * \mathbf{I}_2) = \mathrm{FT}(\mathbf{I}_1) \cdot \mathrm{FT}(\mathbf{I}_2), \tag{46}$$

$$\mathrm{FT}(\mathbf{I}_1 \cdot \mathbf{I}_2) = \mathrm{FT}(\mathbf{I}_1) * \mathrm{FT}(\mathbf{I}_2), \tag{47}$$

where the signals can be feature channels and convolutional kernels. Fourier domain convolutions, $\mathbf{I}_1 * \mathbf{I}_2 = \mathrm{FT}^{-1}(\mathrm{FT}(\mathbf{I}_1) \cdot \mathrm{FT}(\mathbf{I}_2))$, are increasingly efficient, relative to signal domain convolutions, as kernel and image sizes increase [1114]. Indeed, Fourier domain convolutions are exploited to enable faster training with large kernels in Fourier CNNs [1114, 1116]. However, Fourier CNNs are rare as most researchers use small 3×3 kernels, following University of Oxford Visual Geometry Group (VGG) CNNs [1117].

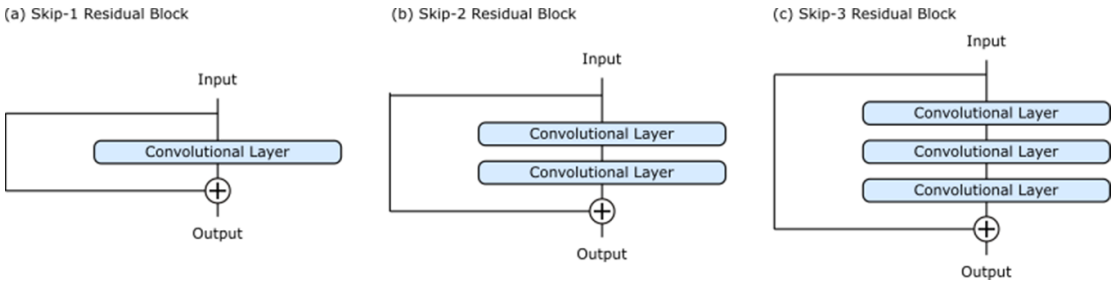

Figure 10. Residual blocks where (a) one, (b) two, and (c) three convolutional layers are skipped. Typically, convolutional layers are followed by batch normalization then activation.

4.4. Skip connections

Residual connections [1080] add a signal after skipping ANN layers, similar to cortical skip connections [1118, 1119]. Residuals improve DNN performance by preserving gradient norms during backpropagation [537, 1120] and avoiding bad local minima [1121] by smoothing DNN loss landscapes [1122]. In practice, residuals enable DNNs to behave like an ensemble of shallow networks [1123] that learn to iteratively estimate outputs [1124]. Mathematically, a residual layer learns parameters, $\mathbf{w}_l$, of a perturbative function, $f_l(\mathbf{x}_l, \mathbf{w}_l)$, that maps a signal, $\mathbf{x}_l$, at depth l to depth $l+1$,

$$\mathbf{x}_{l+1} = \mathbf{x}_l + f_l(\mathbf{x}_l, \mathbf{w}_l)\,. \tag{48}$$

Residuals were developed for CNNs [1080], and examples of residual connections that skip one, two and three convolutional layers are shown in figure 10. Nonetheless, residuals are also used in MLPs [1125] and RNNs [1126–1128]. Representational capacity of perturbative functions increases as the number of skipped layers increases. As result, most residuals skip two or three layers. Skipping one layer rarely improves performance due to its low representational capacity [1080].

There are a range of residual connection variants that can improve performance. For example, highway networks [1129, 1130] apply a gating function to skip connections, and dense networks [1131–1133] use a high number of residual connections from multiple layers. Another example is applying a 1×1 convolutional layer to x_l before addition [539, 1080] where $f_l(x_l, w_l)$ spatially resizes or changes numbers of feature channels. However, resizing with norm-preserving convolutional layers [1120] before residual blocks can often improve performance. Finally, long additive [1134] residuals that connect DNN inputs to outputs are often applied to DNNs that learn perturbative functions.

A limitation of preserving signal information with residuals [1135, 1136] is that residuals make DNNs learn perturbative functions, which can limit accuracy of DNNs that learn non-perturbative functions if they do not have many layers. Feature channel concatenation is an alternative approach that not perturbative, and that supports combination of layers with different numbers of feature channels. In encoder-decoders, a typical example is concatenating features computed near the start with layers near the end to help resolve output features [305, 306, 308, 316]. Concatenation can also combine embeddings of different [1137, 1138] or variants of [366] input features by multiple DNNs. Finally, peephole connections in RNNs can improve performance by using concatenation to combine cell state information with other cell inputs [1139, 1140].

5. Architecture

There is a high variety of ANN architectures [4–7] that are trained to minimize losses for a range of applications. Many of the most popular ANNs are also the simplest, and information about them is readily available. For example, encoder-decoder [305–308, 502–504] or classifier [272] ANNs usually consist of single feedforward sequences of layers that map inputs to outputs. This section introduces more advanced ANNs used in electron microscopy, including actor-critics, GANs, RNNs, and variational autoencoders (VAEs). These ANNs share weights between layers or consist of multiple subnetworks. Other notable architectures include recursive CNNs [1078, 1079], network-in-networks [1141], and transformers [1142, 1143]. Although they will not be detailed in this review, their references may be good starting points for research.

5.1. Actor-critic

Most ANNs are trained by gradient descent using backpropagated gradients of a differentiable loss function cf. section 6.1. However, some losses are not differentiable. Examples include losses of actors directing their

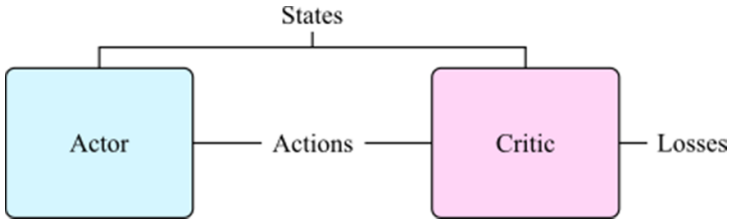

Figure 11. Actor-critic architecture. An actor outputs actions based on input states. A critic then evaluates action-state pairs to predict losses.

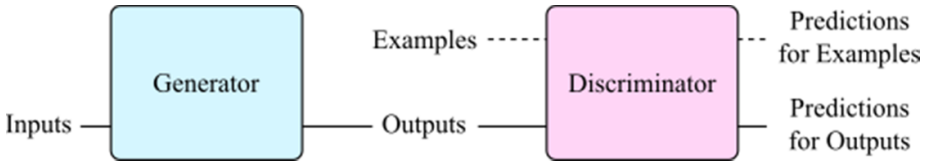

Figure 12. Generative adversarial network architecture. A generator learns to produce outputs that look realistic to a discriminator, which learns to predict whether examples are real or generated.

vision [1144, 1145], and playing competitive [24] or score-based [1146, 1147] computer games. To overcome this limitation, a critic [1148] can be trained to predict differentiable losses from action and state information, as shown in figure 11. If the critic does not depend on states, it is a surrogate loss function [1149, 1150]. Surrogates are often fully trained before actor optimization, whereas critics that depend on actor-state pairs are often trained alongside actors to minimize the impact of catastrophic forgetting [1151] by adapting to changing actor policies and experiences. Alternatively, critics can be trained with features output by intermediate layers of actors to generate synthetic gradients for backpropagation [1152].

5.2. Generative adversarial network

GANs [197–200] consist of generator and discriminator subnetworks that play an adversarial game, as shown in figure 12. Generators learn to generate outputs that look realistic to discriminators, whereas discriminators learn to predict whether examples are real or generated. Most GANs are developed to generate visual media with realistic characteristics. For example, partial STEM images infilled with a GAN are less blurry than images infilled with a non-adversarial generator trained to minimize MSEs [201]; cf. figure 2. Alternatively, computationally inexpensive loss functions designed by humans, such as structural similarity index measures [1153] and Sobel losses [231], can improve generated output realism. However, it follows from the universal approximator theorems [37–45] that training with ANN discriminators can often yield more realistic outputs.

There are many popular GAN loss functions and regularization mechanisms [1154–1158]. Traditionally, GANs were trained to minimize logarithmic discriminator, D, and generator, G, losses [1159],

$$L_D = -\log D(\mathbf{x}) - \log(1 - D(G(\mathbf{z}))), \tag{49}$$

$$L_G = \log(1 - D(G(\mathbf{z}))), \tag{50}$$

where $\mathbf{z}$ are generator inputs, $G(\mathbf{z})$ are generated outputs, and $\mathbf{x}$ are example outputs. Discriminators predict labels, $D(\mathbf{x})$ and $D(G(\mathbf{z}))$, where target labels are 0 and 1 for generated and real examples, respectively. Limitedly, logarithmic losses are numerically unstable for $D(\mathbf{x}) \to 0$ or $D(G(\mathbf{z})) \to 1$, as the denominator, $f(x)$, in $\partial_x \log f(x) = \partial_x f(x)/f(x)$ vanishes. In addition, discriminators must be limited to $D(\mathbf{x}) > 0$ and $D(G(\mathbf{z})) < 1$, so that logarithms are not complex. To avoid these issues, we recommend training discriminators with squared difference losses [1160, 1161],

$$L_D = (D(\mathbf{x}) - 1)^2 + D(G(\mathbf{z}))^2, \tag{51}$$

$$L_G = (D(G(\mathbf{z})) - 1)^2. \tag{52}$$

However, there are a variety of other alternatives to logarithmic loss functions that are also effective [1154, 1155].

A variety of methods have been developed to improve GAN training [995, 1162]. The most common issues are catastrophic forgetting [1151] of previous learning, and mode collapse [1163] where generators only output examples for a subset of a target domain. Mode collapse often follows discriminators becoming Lipschitz discontinuous. Wasserstein GANs [1164] avoid mode collapse by clipping trainable variables, albeit often at the cost of 5–10 discriminator training iterations per generator training iteration. Alternatively, Lipschitz continuity can be imposed by adding a gradient penalty [1165] to GAN losses, such as differences of L2 norms of discriminator gradients from unity,

$$\tilde{x} = G(\mathbf{z}), \tag{53}$$

$$\hat{\mathbf{x}} = \epsilon\mathbf{x} + (1-\epsilon)\tilde{\mathbf{x}}, \tag{54}$$

$$L_D = D(\tilde{\mathbf{x}}) - D(\mathbf{x}) + \lambda(||\partial_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})||_2 - 1)^2, \tag{55}$$

$$L_G = -D(G(\mathbf{z})), \tag{56}$$

where $\varepsilon \in [0, 1]$ is a uniform random variate, λ weights the gradient penalty, and $\tilde{\mathbf{x}}$ is an attempt to generate x. However, using a gradient penalty introduces additional gradient backpropagation that increases discriminator training time. There are also a variety of computationally inexpensive tricks that can improve training, such as adding noise to labels [995, 1075, 1166] or balancing discriminator and generator learning rates [349]. These tricks can help to avoid discontinuities in discriminator output distributions that can lead to mode collapse; however, we observe that these tricks do not reliably stabilize GAN training.

Instead, we observe that spectral normalization [1017] reliably stabilizes GAN discriminator training in our electron microscopy research [201, 202, 349]. Spectral normalization controls Lipschitz constants of discriminators by fixing the spectral norms of their weights, as introduced in section 4.2. Advantages of spectral normalization include implementations based on the power iteration method [1023, 1024] being computationally inexpensive, not adding a regularizing loss function that could detrimentally compete [1167, 1168] with discrimination losses, and being effective with one discriminator training iterations per generator training iteration [1017, 1169]. Spectral normalization is popular in GANs for high-resolution image synthesis, where it is also applied in generators to stabilize training [1170].

There are a variety of GAN architectures [1171]. For high-resolution image synthesis, computation can be decreased by training multiple discriminators to examine image patches at different scales [201, 1172]. For domain translation characterized by textural differences, a cyclic GAN [1004, 1173] consisting of two GANs can map from one domain to the other and vice versa. Alternatively, two GANs can share intermediate layers to translate inputs via a shared embedding domain [1174]. Cyclic GANs can also be combined with a siamese network [279–281] for domain translation beyond textural differences [1175]. Finally, discriminators can introduce auxiliary losses to train DNNs to generalize to examples from unseen domains [1176–1178].

5.3. Recurrent neural network

RNNs [531–536] reuse an ANN cell to process each step of a sequence. Most RNNs learn to model long-term dependencies by gradient backpropagation through time [1179]. The ability of RNNs to utilize past experiences enables them to model partially observed and variable length Markov decision processes [1180, 1181] (MDPs). Applications of RNNs include directing vision [1144, 1145], image captioning [1182, 1183], language translation [1184], medicine [77], natural language processing [1185, 1186], playing computer games [24], text classification [1055], and traffic forecasting [1187]. Many RNNs are combined with CNNs to embed visual media [1145] or words [1188, 1189], or to process RNN outputs [1190, 1191]. RNNs can also be combined with MLPs [1144], or text embeddings [1192] such as BERT [1192, 1193], continuous bag-of-words [1194–1196], doc2vec [1197, 1198], GloVe [1199], and word2vec [1194, 1200].

The most popular RNNs consist of long short-term memory [1201–1204] (LSTM) cells or gated recurrent units [1202, 1205–1207] (GRUs). LSTMs and GRUs are popular as they solve the vanishing gradient problem [537, 1208, 1209] and have consistently high performance [1210–1215]. Their architectures are shown in figure 13. At step t, an LSTM outputs a hidden state, h_t, and cell state, C_t, given by

$$\mathbf{f}_t = \sigma(\mathbf{w}_f \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f), \tag{57}$$

$$\mathbf{i}_t = \sigma(\mathbf{w}_i \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i), \tag{58}$$

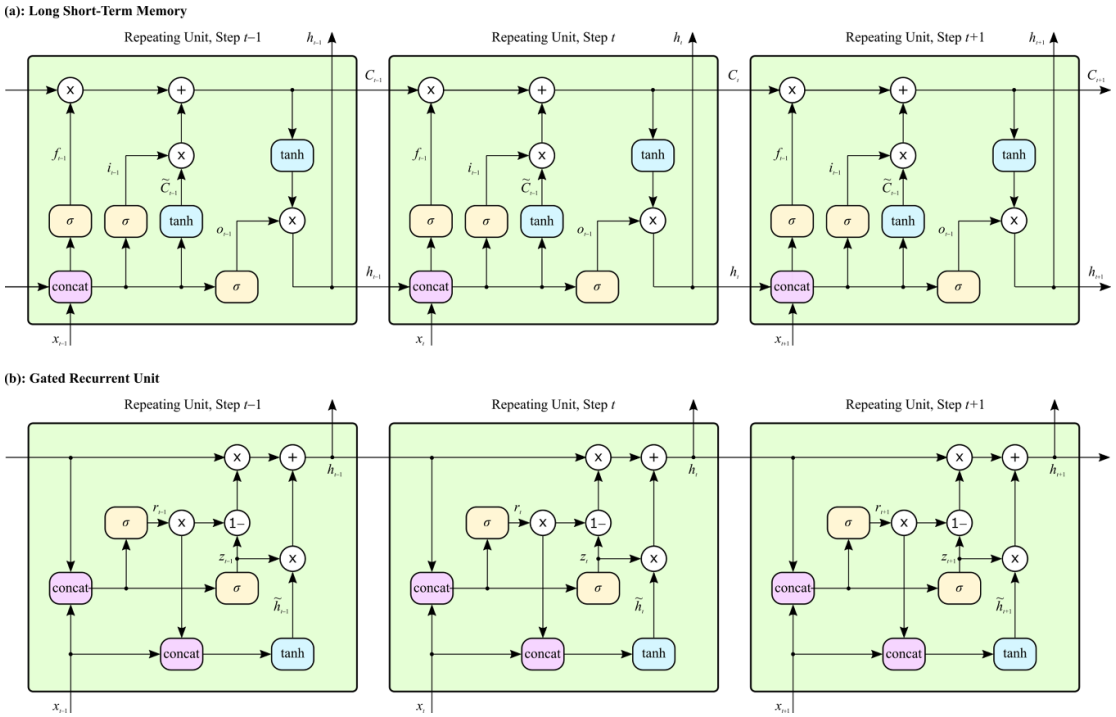

Figure 13. Architectures of recurrent neural networks with (a) long short-term memory (LSTM) cells, and (b) gated recurrent units (GRUs).

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{w}_C \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_C)\,, \tag{59}$$

$$\mathbf{C}_t = \mathbf{f}_t \mathbf{C}_{t-1} + \mathbf{i}_t \tilde{\mathbf{C}}_t\,, \tag{60}$$

$$\mathbf{o}_t = \sigma(\mathbf{w}_o \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_o)\,, \tag{61}$$

$$\mathbf{h}_t = \mathbf{o}_t \tanh(\mathbf{C}_t)\,, \tag{62}$$

where $\mathbf{C}_{t-1}$ is the previous cell state, $\mathbf{h}_{t-1}$ is the previous hidden state, $\mathbf{x}_t$ is the step input, and σ is a logistic sigmoid function of equation (10*a*), $[\mathbf{x}, \mathbf{y}]$ is the concatenation of $\mathbf{x}$ and $\mathbf{y}$ channels, and $(\mathbf{w}_f, \mathbf{b}_f)$, $(\mathbf{w}_i, \mathbf{b}_i)$, $(\mathbf{w}_C, \mathbf{b}_C)$ and $(\mathbf{w}_o, \mathbf{b}_o)$ are pairs of weights and biases. A GRU performs fewer computations than an LSTM and does not have separate cell and hidden states,

$$\mathbf{z}_t = \sigma(\mathbf{w}_z \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_z)\,, \tag{63}$$

$$\mathbf{r}_t = \sigma(\mathbf{w}_r \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_r)\,, \tag{64}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{w}_h \cdot [\mathbf{r}_t \mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_h)\,, \tag{65}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t)\mathbf{h}_{t-1} + \mathbf{z}_t \tilde{\mathbf{h}}_t\,, \tag{66}$$

where $(\mathbf{w}_z, \mathbf{b}_z)$, $(\mathbf{w}_r, \mathbf{b}_r)$, and $(\mathbf{w}_h, \mathbf{b}_h)$ are pairs of weights and biases. Minimal gated units can further reduce computation [1216]. A large-scale analysis of RNN architectures for language translation found that LSTMs consistently outperform GRUs [1210]. GRUs struggle with simple languages that are learnable by LSTMs as the combined hidden and cell states of GRUs make it more difficult for GRUs to perform unbounded counting [1214]. However, further investigations found that GRUs can outperform LSTMs on tasks other than language translation [1211], and that GRUs can outperform LSTMs on some datasets [1212, 1213, 1217]. Overall, LSTM performance is usually comparable to that of GRUs.

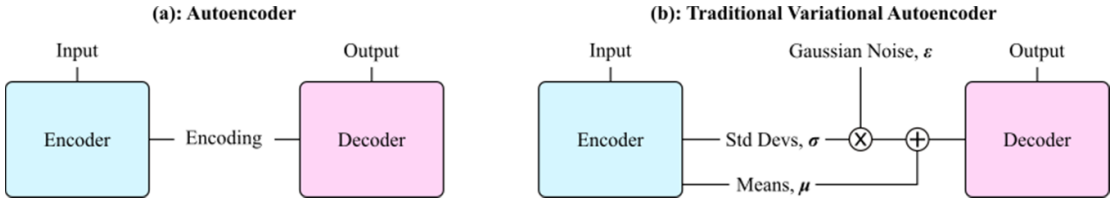

Figure 14. Architectures of autoencoders where an encoder maps an input to a latent space and a decoder learns to reconstruct the input from the latent space. (a) An autoencoder encodes an input in a deterministic latent space, whereas a (b) traditional variational autoencoder encodes an input as means, μ, and standard deviations, σ, of Gaussian multivariates, $\mu + \sigma \cdot \varepsilon$, where ε is a standard normal multivariate.

There are a variety of alternatives to LSTM and GRUs. Examples include continuous time RNNs [1218–1222], Elman [1223] and Jordan [1224] networks, independently RNNs [1225], Hopfield networks [1226], recurrent MLPs [1227]. However, none of the variants offer consistent performance benefits over LSTMs for general sequence modelling. Similarly, augmenting LSTMs with additional connections, such as peepholes [1139, 1140] and projection layers [1228], does not consistently improve performance. For electron microscopy, we recommend defaulting to LSTMs as we observe that their performance is more consistently high than performance of other RNNs. However, LSTM and GRU performance is often comparable, so GRUs are also a good choice to reduce computation.

There are a variety of architectures based on RNNs. Popular examples include deep RNNs [1229] that stack RNN cells to increase representational ability, bidirectional RNNs [1230–1233] that process sequences both forwards and in reverse to improve input utilization, and using separate encoder and decoder subnetworks [1205, 1234] to embed inputs and generate outputs. Hierarchical RNNs [1235–1239] are more complex models that stack RNNs to efficiently exploit hierarchical sequence information, and include multiple timescale RNNs [1240, 1241] that operate at multiple sequence length scales. Finally, RNNs can be augmented with additional functionality to enable new capabilities. For example, attention [1182, 1242–1244] mechanisms can enable more efficient input utilization. Further, creating a neural Turing machine by augmenting a RNN with dynamic external memory [1245, 1246] can make it easier for an agent to solve dynamic graphs.

5.4. Autoencoders

Autoencoders [1247–1249] (AEs) learn to efficiently encode inputs, $\mathbf{I}$, without supervision. An AE consists of a encoder, E, and decoder, D, as shown in figure 14(a). Most encoders and decoders are jointly trained [1250] to restore inputs from encodings, $E(\mathbf{I})$, to minimize a MSE loss,

$$L_{\text{AE}} = \text{MSE}(D(E(\mathbf{I})), \mathbf{I}), \tag{67}$$

by gradient descent. In practice, DNN encoders and decoders yield better compression [1248] than linear techniques, such as PCA [1251], or shallow ANNs. Indeed, deep AEs can outperform JPEG image compression [1252]. Denoising autoencoders (DAEs) [1253–1257] are a popular AE variant that can learn to remove artefacts by artificially corrupting inputs inside encoders. Alternatively, contractive autoencoders [1258, 1259] can decrease sensitivity to input values by adding a loss to minimize gradients w.r.t. inputs. Most DNNs that improve electron micrograph signal-to-noise are DAEs.

In general, semantics of AE outputs are pathological functions of encodings. To generate outputs with well-behaved semantics, traditional VAEs [969, 1260, 1261] learn to encode means, $\boldsymbol{\mu}$, and standard deviations, $\boldsymbol{\sigma}$, of Gaussian multivariates. Meanwhile, decoders learn to reconstruct inputs from sampled multivariates, $\boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a standard normal multivariate. Traditional VAE architecture is shown in figure 14(b). Usually, VAE encodings are regularized by adding Kullback–Leibler (KL) divergence of encodings from standard multinormals to an AE loss function,

$$L_{\text{VAE}} = \text{MSE}(D(\boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}), \mathbf{I}) + \frac{\lambda_{\text{KL}}}{2Bu} \sum_{i=1}^{B} \sum_{j=1}^{u} \mu_{ij}^2 + \sigma_{ij}^2 - \log(\sigma_{ij}^2) - 1, \tag{68}$$

where λ_{KL} weights the contribution of the KL divergence loss for a batch size of B, and a latent space with u degrees of freedom. However, variants of Gaussian regularization can improve clustering [231], and sparse autoencoders [1262–1265] that regularize encoding sparsity can encode more meaningful features. To generate realistic outputs, a VAE can be combined with a GAN to create a VAE-GAN [1266–1268]. Adding a

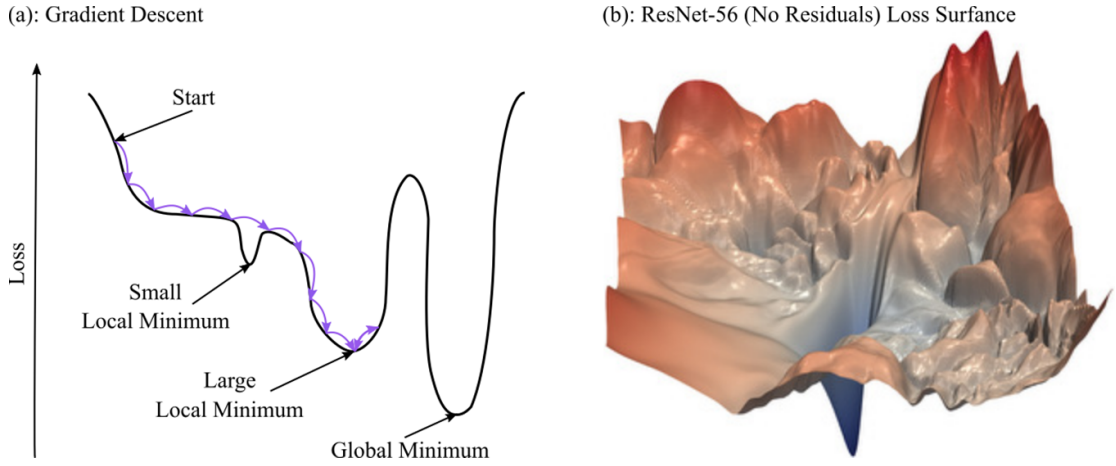

Figure 15. Gradient descent. (a) Arrows depict steps across one dimension of a loss landscape as a model is optimized by gradient descent. In this example, the optimizer traverses a small local minimum; however, it then gets trapped in a larger sub-optimal local minimum, rather than reaching the global minimum. (b) Experimental DNN loss surface for two random directions in parameter space showing many local minima [1122]. The image in part (b) is reproduced with permission under an MIT license [1291].

Algorithm 1. Optimization by gradient descent.

Initialize a model, $f(\mathbf{x})$, with trainable parameters, $\boldsymbol{\theta}_1$.
for training step $t = 1, T$ **do**
  Forwards propagate a randomly sampled batch of inputs, $\mathbf{x}$, through the model to compute outputs, $\mathbf{y} = f(\mathbf{x})$.
  Compute loss, L_t, for outputs.
  Use the differentiation chain rule [1292] to backpropagate gradients of the loss to trainable parameters, $\boldsymbol{\theta}_{t-1}$.
  Apply an optimizer to the gradients to update $\boldsymbol{\theta}_{t-1}$ to $\boldsymbol{\theta}_t$.
end for

loss to minimize differences between gradients of generated and target outputs is computationally inexpensive alternative that can generate realistic outputs for some applications [231].

A popular application of VAEs is data clustering. For example, VAEs can encode hash tables [1269–1273] for search engines, and we use VAEs as the basis of our electron micrograph search engines [231]. Encoding clusters visualized by tSNE can be labelled to classify data [231], and encoding deviations from clusters can be used for anomaly detection [1274–1278]. In addition, learning encodings with well-behaved semantics enables encodings to be used for semantic manipulation [1278, 1279]. Finally, VAEs can be used as generative models to create synthetic populations [1280, 1281], develop new chemicals [1282–1285], and synthesize underrepresented data to reduce imbalanced learning [1286].

6. Optimization

Training, testing, deployment and maintenance of machine learning systems is often time-consuming and expensive [1287–1290]. The first step is usually preparing training data and setting up data pipelines for ANN training and evaluation. Typically, ANN parameters are randomly initialized for optimization by gradient descent, possibly as part of an automatic machine learning (autoML) algorithm. RL is a special optimization case where the loss is a discounted future reward. During training, ANN components are often regularized to stabilize training, accelerate convergence, or improve performance. Finally, trained models can be streamlined for efficient deployment. This section introduces each step. We find that electron microscopists can be apprehensive about robustness and interpretability of ANNs, so we also provide subsections on model evaluation and interpretation.

6.1. Gradient descent

Most ANNs are iteratively trained by gradient descent [465, 1303–1307], as described by algorithm 1 and shown in figure 15. To minimize computation, results at intermediate stages of forward propagation, where inputs are mapped to outputs, are often stored in memory. Storing the forwards pass in memory enables backpropagation memoization by sequentially computing gradients w.r.t. trainable parameters. To reduce memory costs for large ANNs, a subset of intermediate forwards pass results can be saved as starting points to recompute other stages during backpropagation [1308, 1309]. Alternatively, forward pass computations

Vanilla SGD [1293, 1294] $[\eta]$

$$\theta_t = \theta_{t-1} - \eta\partial_\theta L_t \tag{69}$$

Momentum [1295] $[\eta, \gamma]$

$$v_t = \gamma v_{t-1} + \eta\partial_\theta L_t \tag{70}$$
$$\theta_t = \theta_{t-1} - v_t \tag{71}$$

Nesterov momentum [1296–1298] $[\eta, \gamma]$

$$\phi = \theta_{t-1} + \eta\gamma v_{t-1} \tag{72}$$
$$v_t = \gamma v_{t-1} + \partial_\theta L_t \tag{73}$$
$$\theta_t = \phi - \eta v_t(1+\gamma) \tag{74}$$

Quasi-hyperbolic momentum [1299] $[\eta, \beta, \nu]$

$$g_t = \beta g_{t-1} + (1-\beta)\partial_\theta L_t \tag{75}$$
$$\theta_t = \theta_{t-1} - \eta(v g_t + (1-v)\partial_\theta L_t) \tag{76}$$

AggMo [1300] $[\eta, \beta^{(1)}, ..., \beta^{(K)}]$

$$v_t^{(i)} = \beta^{(i)} v_{t-1}^{(i)} - (\partial_\theta L_t) \tag{77}$$
$$\theta_t = \theta_{t-1} + \frac{\eta}{K}\sum_{i=1}^{K} v_t^{(i)} \tag{78}$$

RMSProp [1301] $[\eta, \beta, \epsilon]$

$$v_t = \beta v_{t-1} + (1-\beta)(\partial_\theta L_t)^2 \tag{79}$$
$$\theta_t = \theta_{t-1} - \frac{\eta}{(v_t+\epsilon)^{1/2}}\partial_\theta L_t \tag{80}$$

ADAM [1302] $[\eta, \beta_1, \beta_2, \epsilon]$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\partial_\theta L_t \tag{81}$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\partial_\theta L_t)^2 \tag{82}$$
$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{83}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{84}$$
$$\theta_t = \theta_{t-1} - \frac{\eta}{\hat{v}_t^{1/2}+\epsilon}\hat{m}_t \tag{85}$$

AdaMax [1302] $[\eta, \beta_1, \beta_2]$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\partial_\theta L_t \tag{86}$$
$$u_t = \max(\beta_2 u_{t-1}, |\partial_\theta L_t|) \tag{87}$$
$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{88}$$
$$\theta_t = \theta_{t-1} - \frac{\eta}{u_t}\hat{m}_t \tag{89}$$

Figure 16. Update rules of various gradient descent optimizers for a trainable parameter, θ_t, at iteration t, gradients of losses w.r.t. the parameter, $\partial_\theta L_t$, and learning rate, η. Hyperparameters are listed in square brackets.

can be split across multiple devices [1310]. Optimization by gradient descent plausibly models learning in some biological systems [1311]. However, gradient descent is not generally an accurate model of biological learning [1312–1314].

There are many popular gradient descent optimizers for deep learning [1303–1305]. Update rules for eight popular optimizers are summarized in figure 16. Other optimizers include AdaBound [1315], AMSBound [1315], AMSGrad [1316], Lookahead [1317], NADAM [1318], Nostalgic Adam [1319], Power Gradient Descent [1320], Rectified ADAM [1321], and trainable optimizers [1322–1326]. Gradient descent is effective in the high-dimensional optimization spaces of overparameterized ANNs [1327] as the probability of getting trapped in a sub-optimal local minima decreases as the number of dimensions increases. The simplest optimizer is 'vanilla' stochastic gradient descent (SGD), where a trainable parameter perturbation, $\Delta\theta_t = \theta_t - \theta_{t-1}$, is the product of a learning rate, η, and derivative of a loss, L_t, w.r.t. the trainable parameter, $\partial_\theta L_t$. However, vanilla SGD convergence is often limited by unstable parameter oscillations as it a low-order local optimization method [1328]. Further, vanilla SGD has no mechanism to adapt to varying gradient sizes, which vary effective learning rates as $\Delta\theta \propto \partial_\theta L_t$.

To accelerate convergence, many optimizers introduce a momentum term that weights an average of gradients with past gradients [1296, 1329, 1330]. Momentum-based optimizers in figure 16 are momentum, Nesterov momentum [1296, 1297], quasi-hyperbolic momentum [1299], AggMo [1300], ADAM [1302], and AdaMax [1302]. To standardize effective learning rates for every layer, adaptive optimizers normalize updates based on an average of past gradient sizes. Adaptive optimizers in figure 16 are RMSProp [1301], ADAM [1302], and AdaMax [1302], which usually result in faster convergence and higher accuracy than other optimizers [1331, 1332]. However, adaptive optimizers can be outperformed by vanilla SGD due to overfitting [1333], so some researchers adapt adaptive learning rates to their variance [1321] or transition from adaptive optimization to vanilla SGD as training progresses [1315]. For electron microscopy we recommend adaptive optimization with Nadam [1318], which combines ADAM with Nesterov momentum, as it is well-established and a comparative analysis of select gradient descent optimizers found that it often achieves higher performance than other popular optimizers [1334]. Limitingly, most adaptive optimizers slowly adapt to changing gradient sizes e.g. a default value for ADAM β_2 is 0.999 [1302]. To prevent learning being destabilized by spikes in gradient sizes, adaptive optimizers can be combined with adaptive learning rate [261, 1315] or gradient [1208, 1335, 1336] clipping.

For non-adaptive optimizers, effective learning rates are likely to vary due to varying magnitudes of gradients w.r.t. trainable parameters. Similarly, learning by biological neurons varies as stimuli usually activate a subset of neurons [1337]. However, all neuron outputs are usually computed for ANNs. Thus, not effectively using all weights to inform decisions is computational inefficient. Further, inefficient weight updates can limit representation capacity, slow convergence, and decrease training stability. A typical example is effective learning rates varying between layers. Following the chain rule, gradients backpropagated to the ith layer of a DNN from its start are

$$\frac{\partial L_t}{\partial \mathbf{x}_i} = \left(\prod_{l=i}^{L-1} \frac{\partial \mathbf{x}_{l+1}}{\partial \mathbf{x}_l} \right) \frac{\partial L_t}{\partial \mathbf{x}_L}, \tag{90}$$

for a DNN with L layers. Vanishing gradients [537, 1208, 1209] occur when many layers have $\partial x_{l+1}/\partial x_l \ll 1$. For example, DNNs with logistic sigmoid activations often exhibit vanishing gradients as their maximum gradient is 1/4; cf. equation (10*b*). Similarly, exploding gradients [537, 1208, 1209] occur when many layers have $\partial x_{l+1}/\partial x_l \gg 1$. Adaptive optimizers alleviate vanishing and exploding gradients by dividing gradients by their expected sizes. Nevertheless, it is essential to combine adaptive optimizers with appropriate initialization and architecture to avoid numerical instability.

Optimizers have a myriad of hyperparameters to be initialized and varied throughout training to optimize performance [1338] cf. figure 16. For example, stepwise exponentially decayed learning rates are often theoretically optimal [1339]. There are also various heuristics that are often effective, such as using a DEMON decay schedule for an ADAM first moment of the momentum decay rate [1340],

$$\beta_1 = \frac{1 - t/T}{(1 - \beta_{\text{init}}) + \beta_{\text{init}}(1 - t/T)} \beta_{\text{init}}, \tag{91}$$

where β_{init} is the initial value of β_1, t is the iteration number, and T is the final iteration number. Developers often optimize ANN hyperparameters by experimenting with a range of heuristic values. Hyperparameter optimization algorithms [1341–1346] can automate optimizer hyperparameter selection. However, automatic hyperparameter optimizers may not yield sufficient performance improvements relative to well-established heuristics to justify their use, especially in initial stages of development.

Alternatives to gradient descent [1347] are rarely used for parameter optimization as they are not known to consistently improve upon gradient descent. For example, simulated annealing [1348, 1349] has been applied to CNN training [1350, 1351], and can be augmented with momentum to accelerate convergence in deep learning [1352]. Simulated annealing can also augment gradient descent to improve performance [1353]. Other approaches include evolutionary [1354, 1355] and genetic [1356, 1357] algorithms, which can be a competitive alternative to deep RL where convergence is slow [1358]. Indeed, recent genetic algorithms have outperformed a popular deep RL algorithm [1359]. Another direction is to augment genetic algorithms with ANNs to accelerate convergence [1360–1363]. Other alternatives to backpropagation include direct search [1364], the Moore–Penrose pseudo inverse [1365]; particle swarm optimization [1366–1369]; and echo-state networks [1370–1372] and extreme learning machines [1373–1379], where some randomly initialized weights are never updated.

6.2. Reinforcement learning

RL [1380–1386] is where a machine learning system, or 'actor', is trained to perform a sequence of actions. Applications include autonomous driving [1387–1389], communications network control [1390, 1391], energy and environmental management [1392, 1393], playing games [24–29, 1146, 1394], and robotic manipulation [1395, 1396]. To optimize a MDP [1180, 1181], a discounted future reward, Q_t, at step t in a MDP with T steps is usually calculated from step rewards, r_t, with Bellman's equation,

$$Q_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}, \tag{92}$$

where $\gamma \in [0, 1)$ discounts future step rewards. To be clear, multiplying Q_t by -1 yields a loss that can be minimized using the methods in section 6.1.

In practice, many MDPs are partially observed or have non-differentiable losses that may make it difficult to learn a good policy from individual observations. However, RNNs can often learn a model of their environments from sequences of observations [1147]. Alternatively, FNNs can be trained with groups of observations that contain more information than individual observations [1146, 1394]. If losses are not differentiable, a critic can learn to predict differentiable losses for actor training cf. section 5.1. Alternatively,

actions can be sampled from a differentiable probability distribution [1144, 1397] as training losses given by products of losses and sampling probabilities are differentiable. There are also a variety of alternatives to gradient descent introduced at the end of section 6.1 that do not require differentiable loss functions.

There are a variety of exploration strategies for RL [1398, 1399]. Adding Ornstein–Uhlenbeck [1400] (OU) noise to actions is effective for continuous control tasks optimized by deep deterministic policy gradients [1146] or recurrent deterministic policy gradients [1147] RL algorithms. Adding Gaussian noise achieves similar performance for optimization by TD3 [1401] or D4PG [1402] RL algorithms. However, a comparison of OU and Gaussian noise across a variety of tasks [1403] found that OU noise usually achieves similar performance to or outperforms Gaussian noise. Similarly, exploration can be induced by adding noise to ANN parameters [1404, 1405]. Other approaches to exploration include rewarding actors for increasing action entropy [1405–1407] and intrinsic motivation [1408–1410], where ANNs are incentified to explore actions that they are unsure about.

RL algorithms are often partitioned into online learning [1411, 1412], where training data is used as it is acquired; and offline learning [1413, 1414], where a static training dataset has already been acquired. However, many algorithms operate in an intermediate regime, where data collected with an online policy is stored in an experience replay [1415–1417] buffer for offline learning. Training data is often sampled at random from a replay. However, prioritizing the replay of data with high losses [993] or data that results in high policy improvements [992] often improves actor performance. A default replay buffer size of around 10^6 examples is often used; however, training is sensitive to replay buffer size [1418]. If the replay is too small, changes in actor policy may destabilize training; whereas if the replay is too large, convergence may be slowed by delays before the actor learns from policy changes.

6.3. Automatic machine learning

There are a variety of AutoML [1419–1423] algorithms that can create and optimize ANN architectures and learning policies for a dataset of input and target output pairs. Most AutoML algorithms are based on RL or evolutionary algorithms. Examples of AutoML algorithms include AdaNet [1424, 1425], Auto-DeepLab [1426], AutoGAN [1427], Auto-Keras [1428], auto-sklearn [1429], DARTS+ [1430], EvoCNN [271], H2O [1431], Ludwig [1432], MENNDL [1433, 1434], NASBOT [1435], XNAS [1436], and others [1437–1441]. AutoML is becoming increasingly popular as it can achieve higher performance than human developers [1077, 1442] and enables human developer time to be traded for potentially cheaper computer time. Nevertheless, AutoML is currently limited to established ANN architectures and learning policies. Following, we recommend that researchers either focus on novel ANN architectures and learning policies or developing ANNs for novel applications.

6.4. Initialization

How ANN trainable parameters are initialized [537, 1443] is related to model capacity [1444]. Further, initializing parameters with values that are too small or large can cause slow learning or divergence [537]. Careful initialization can also prevent training by gradient descent being destabilized by vanishing or exploding gradients [537, 1208, 1209], or high variance of length scales across layers [537]. Finally, careful initialization can enable momentum to accelerate convergence and improve performance [1296]. Most trainable parameters are multiplicative weights or additive biases. Initializing parameters with constant values would result in every parameter in a layer receiving the same updates by gradient descent, reducing model capacity. Thus, weights are often randomly initialized. Following, biases are often initialized with constant values due to symmetry breaking by the weights.

Consider the projection of n_{in} inputs, $\mathbf{x}^{\text{input}} = \{x_1^{\text{input}}, \ldots, x_{n_{\text{in}}}^{\text{input}}\}$, to n_{out} outputs, $\mathbf{x}^{\text{output}} = \{x_1^{\text{output}}, \ldots, x_{n_{\text{out}}}^{\text{output}}\}$, by an $n_{\text{in}} \times n_{\text{out}}$ weight matrix, $\mathbf{w}$. The expected variance of an output element is [1443]

$$\text{Var}(\mathbf{x}^{\text{output}}) = n_{\text{in}}\text{E}(\mathbf{x}^{\text{input}})^2\text{Var}(\mathbf{w}) + n_{\text{in}}\text{E}(\mathbf{w})^2\text{Var}(\mathbf{x}^{\text{input}}) + n_{\text{in}}\text{Var}(\mathbf{w})\text{Var}(\mathbf{x}^{\text{input}}), \tag{93}$$

where $\text{E}(\mathbf{x})$ and $\text{Var}(\mathbf{x})$ denote the expected mean and variance of elements of $\mathbf{x}$, respectively. For similar length scales across layers, $\text{Var}(\mathbf{x}^{\text{output}})$ should be constant. Initially, similar variances can be achieved by normalizing ANN inputs to have zero mean, so that $\text{E}(\mathbf{x}^{\text{input}}) = 0$, and initializing weights so that $\text{E}(\mathbf{w}) = 0$ and $\text{Var}(\mathbf{w}) = 1/n_{\text{in}}$. However, parameters can shift during training, destabilizing learning. To compensate for parameter shift, popular normalization layers like batch normalization often impose $\text{E}(\mathbf{x}^{\text{input}}) = 0$ and $\text{Var}(\mathbf{x}^{\text{input}}) = 1$, relaxing need for $\text{E}(\mathbf{x}^{\text{input}}) = 0$ or $\text{E}(\mathbf{w}) = 0$. Nevertheless, training will still be sensitive to the length scale of trainable parameters.

There are a variety of popular weight initializers that adapt weights to ANN architecture. One of the oldest methods is LeCun initialization [941, 951], where weights are initialized with variance,

$$\text{Var}(\mathbf{w}) = \frac{1}{n_{\text{in}}}, \tag{94}$$

which is argued to produce outputs with similar length scales in the previous paragraph. However, a similar argument can be made for initializing with $\mathrm{Var}(\mathbf{w}) = 1/n_{\text{out}}$ to produce similar gradients at each layer during the backwards pass [1443]. As a compromise, Xavier initialization [1445] computes an average:

$$\mathrm{Var}(\mathbf{w}) = \frac{2}{n_{\text{in}} + n_{\text{out}}}. \tag{95}$$

However, adjusting weights for n_{out} is not necessary for adaptive optimizers like ADAM, which divide gradients by their length scales, unless gradients will vanish or explode. Finally, He initialization [22] doubles the variance of weights to

$$\mathrm{Var}(\mathbf{w}) = \frac{2}{n_{\text{in}}}, \tag{96}$$

and is often used in ReLU networks to compensate for activation functions halving variances of their outputs [22, 1443, 1446]. Most trainable parameters are initialized from either a zero-centred Gaussian or uniform distribution. For convenience, the limits of such a uniform distribution are $\pm(3\mathrm{Var}(\mathbf{w}))^{1/2}$. Uniform initialization can outperform Gaussian initialization in DNNs due to Gaussian outliers harming learning [1443]. However, issues can be avoided by truncating Gaussian initialization, often to two standard deviations, and rescaling to its original variance.

Some initializers are mainly used for RNNs. For example, orthogonal initialization [1447] often improves RNN training [1448] by reducing susceptibility to vanishing and exploding gradients. Similarly, identity initialization [1449, 1450] can help RNNs to learn long-term dependencies. In most ANNs, biases are initialized with zeros. However, the forget gates of LSTMs are often initialized with ones to decrease forgetting at the start of training [1211]. Finally, the start states of most RNNs are initialized with zeros or other constants. However, random multivariate or trainable variable start states can improve performance [1451].

There are a variety of alternatives to initialization from random multivariates. Weight normalized [1014] ANNs are a popular example of data-dependent initialization, where randomly initialized weight magnitudes and biases are chosen to counteract variances and means of an initial batch of data. Similarly, layer-sequential unit-variance initialization [1452] consists of orthogonal initialization followed by adjusting the magnitudes of weights to counteract variances of an initial batch of data. Other approaches standardize the norms of backpropagated gradients. For example, random walk initialization [1453] finds scales for weights to prevent vanishing or exploding gradients in deep FNNs, albeit with varied success [1452]. Alternatively, MetaInit [1454] scales the magnitudes of randomly initialized weights to minimize changes in backpropagated gradients per iteration of gradient descent.

6.5. Regularization

There are a variety of regularization mechanisms [1455–1458] that modify learning algorithms to improve ANN performance. One of the most popular is LX regularization, which decays weights by adding a loss,

$$L_X = \lambda_X \sum_i \frac{|\theta_i|^X}{X}, \tag{97}$$

weighted by λ_X to each trainable variable, θ_i. L2 regularization [1459–1461] is preferred [1462] for most DNN optimization as subtraction of its gradient, $\partial_{\theta_i} L_2 = \lambda_2 \theta_i$, is equivalent to computationally-efficient multiplicative weight decay. Nevertheless, L1 regularization is better at inducing model sparsity [1463] than L2 regularization, and L1 regularization achieves higher performance in some applications [1464]. Higher performance can also be achieved by adding both L1 and L2 regularization in elastic nets [1465]. LX regularization is most effective at the start of training and becomes less important near convergence [1459]. Finally, L1 and L2 regularization are closely related to lasso [1466] and ridge [1467] regularization, respectively, whereby trainable parameters are adjusted to limit L_1 and L_2 losses.

Gradient clipping [1336, 1468–1470] accelerates learning by limiting large gradients, and is most commonly applied to RNNs. A simple approach is to clip gradient magnitudes to a threshold hyperparameter. However, it is more common to scale gradients, $\mathbf{g}_i$, at layer i if their norm is above a threshold, u, so that [1208, 1469]

$$\mathbf{g}_i \leftarrow \begin{cases} \mathbf{g}_i, & \text{if } ||\mathbf{g}_i||_n \leq u \\ \frac{u}{||\mathbf{g}_i||_n}\mathbf{g}_i, & \text{if } ||\mathbf{g}_i||_n > u \end{cases} \tag{98}$$

where $n = 2$ is often chosen to minimize computation. Similarly, gradients can be clipped if they are above a global norm,

$$g_{\text{norm}} = \left(\sum_{i=1}^{L} ||\mathbf{g}_i||_n^n, \right)^{1/n} \tag{99}$$

computed with gradients at L layers. Scaling gradient norms is often preferable to clipping to a threshold as scaling is akin to adapting layer learning rates and does not affect the directions of gradients. Thresholds for gradient clipping are often set based on average norms of backpropagated gradients during preliminary training [1471]. However, thresholds can also be set automatically and adaptively [1335, 1336]. In addition, adaptive gradient clipping algorithms can skip training iterations if gradient norms are anomalously high [1472], which often indicates an imminent gradient explosion.

Dropout [1473–1477] often reduces overfitting by only using a fraction, p_i, of layer i outputs during training, and multiplying all outputs by p_i for inference. However, dropout often increases training time, can be sensitive to p_i, and sometimes lowers performance [1478]. Improvements to dropout at the structural level, such as applying it to convolutional channels, paths, and layers, rather than random output elements, can improve performance [1479]. For example, DropBlock [1480] improves performance by dropping contiguous regions of feature maps to prevent dropout being trivially circumvented by using spatially correlated neighbouring outputs. Similarly, PatchUp [1481] swaps or mixes contiguous regions with regions for another sample. Dropout is often outperformed by Shakeout [1482, 1483], a modification of dropout that randomly enhances or reverses contributions of outputs to the next layer.

Noise often enhances ANN training by decreasing susceptibility to spurious local minima [1484]. Adding noise to trainable parameters can improve generalization [1485, 1486], or exploration for RL [1404]. Parameter noise is usually additive as it does not change an objective function being learned, whereas multiplicative noise can change the objective [1487]. In addition, noise can be added to inputs [1253, 1488], hidden layers [1158, 1489], generated outputs [1490] or target outputs [995, 1491]. However, adding noise to signals does not always improve performance [1217]. Finally, modifying usual gradient noise [1492] by adding noise to gradients can improve performance [1493]. Typically, additive noise is annealed throughout training, so that that final training is with a noiseless model that will be used for inference.

There are a variety of regularization mechanisms that exploit extra training data. A simple approach is to create extra training examples by data augmentation [1494–1496]. Extra training data can also be curated, or simulated for training by domain adaption [1176–1178]. Alternatively, semi-supervised learning [1497–1502] can generate target outputs for a dataset of unpaired inputs to augment training with a dataset of paired inputs and target outputs. Finally, multitask learning [1503–1507] can improve performance by introducing additional loss functions. For instance, by adding an auxiliary classifier to predict image labels from features generated by intermediate DNN layers [1508–1511]. Losses are often manually balanced; however, their gradients can also be balanced automatically and adaptively [1167, 1168].

6.6. Data pipeline

A data pipeline prepares data to be input to an ANN. Efficient pipelines often parallelize data preparation across multiple CPU cores [1512]. Small datasets can be stored in RAM to decrease data access times, whereas large dataset elements are often loaded from files. Loaded data can then be preprocessed and augmented [1494, 1495, 1513–1515]. For electron micrographs, preprocessing often includes replacing non-finite elements, such as NaN and inf, with finite values; linearly transforming intensities to a common range, such as $[-1, 1]$ or zero mean and unit variance; and performing a random combination of flips and 90° to augment data by a factor of eight [70, 201, 202, 231, 349]. Preprocessed examples can then be combined into batches. Typically, multiple batches that are ready to be input are prefetched and stored in RAM to avoid delays due to fluctuating CPU performance.

To efficiently utilize data, training datasets are often reiterated over for multiple training epochs. Usually, training datasets are reiterated over about 10^2 times. Increasing epochs can maximize utilization of potentially expensive training data; however, increasing epochs can lower performance due to overfitting [1516, 1517] or be too computationally expensive [539]. Naively, batches of data can be randomly sampled with replacement during training by gradient descent. However, convergence can be accelerated by reinitializing a training dataset at the start of each training epoch and randomly sampling data without replacement [1518–1522]. Most modern DLFs, such as TensorFlow, provide efficient and easy-to-use functions to control data sampling [1523].

6.7. Model evaluation

There are a variety of methods for ANN performance evaluation [538]. However, most ANNs are evaluated by 1-fold validation, where a dataset is partitioned into training, validation, and test sets. After ANN optimization with a training set, ability to generalize is measured with a validation set. Multiple validations may be performed for training with early stopping [1516, 1517] or ANN learning policy and architecture selection, so final performance is often measured with a test set to avoid overfitting to the validation set. Most researchers favour using single training, validation, and test sets to simplify standardization of performance benchmarks [231]. However, multiple-fold validation [538] or multiple validation sets [1524] can improve performance characterization. Alternatively, models can be bootstrap aggregated [1525] (bagged) from multiple models trained on different subsets of training data. Bagging is usually applied to random forests [1526–1528] or other lightweight models, and enables model uncertainly to be gauged from the variance of model outputs.

For small datasets, model performance is often sensitive to split of data between training and validation sets [1529]. Increasing training set size usually increases model accuracy, whereas increasing validation set size decreases performance uncertainty. Indeed, a scaling law can be used to estimate an optimal tradeoff [1530] between training and validation set sizes. However, most experimenters follow a Pareto [1531] splitting heuristic. For example, we often use a 75:15:10 training-validation-test split [231]. Heuristic splitting is justified for ANN training with large datasets insofar that sensitivity to splitting ratios decreases with increasing dataset size [2].

6.8. Deployment

If an ANN is deployed [1532–1534] on multiple different devices, such as various electron microscopes, a separate model can be trained for each device [403], Alternatively, a single model can be trained and specialized for different devices to decrease training requirements [1535]. In addition, ANNs can remotely service requests from cloud containers [1536–1538]. Integration of multiple ANNs can be complicated by different servers for different DLFs supporting different backends; however, unified interfaces are available. For example, GraphPipe [1539] provides simple, efficient reference model servers for Tensorflow, Caffe2, and ONNX; a minimalist machine learning transport specification based on FlatBuffers [1540]; and efficient client implementations in Go, Python, and Java. In 2020, most ANNs developed researchers were not deployed. However, we anticipate that deployment will become a more prominent consideration as the role of deep learning in electron microscopy matures.

Most ANNs are optimized for inference by minimizing parameters and operations from training time, like MobileNets [1094]. However, less essential operations can also be pruned after training [1541, 1542]. Another approach is quantization, where ANN bit depths are decreased, often to efficient integer instructions, to increase inference throughput [1543, 1544]. Quantization often decreases performance; however, the amount of quantization can be adapted to ANN components to optimize performance-throughput tradeoffs [1545]. Alternatively, training can be modified to minimize the impact of quantization on performance [1546–1548]. Another approach is to specialize bit manipulation for deep learning. For example, signed brain floating point (bfloat16) often improves accuracy on TPUs by using an 8-bit mantissa and 7-bit exponent, rather than a usual 5-bit mantissa and 10-bit exponent [1549]. Finally, ANNs can be adaptively selected from a set of ANNs based on available resources to balance tradeoff of performance and inference time [1550], similar to image optimization for web applications [1551, 1552].

6.9. Interpretation

We find that some electron microscopists are apprehensive about working with ANNs due to a lack of interpretability, irrespective of rigorous ANN validation. We try to address uncertainty by providing loss visualizations in some of our electron microscopy papers [70, 201, 202]. However, there are a variety of popular approaches to explainable artificial intelligence (XAI) [1554–1560]. One of the most popular approaches to XAI is saliency [1561–1564], where gradients of outputs w.r.t. inputs correlate with their importance. Saliency is often computed by gradient backpropagation [1565–1567]. For example, with Grad-CAM [1568] or its variants [1569–1572]. Alternatively, saliency can be predicted by ANNs [1054, 1573, 1574] or a variety of methods inspired by Grad-CAM [1575–1577]. Applications of saliency include selecting useful features from a model [1578], and locating regions in inputs corresponding to ANN outputs [1579].

There are a variety of other approaches to XAI. For example, feature visualization via optimization [1553, 1580–1583] can find inputs that maximally activate parts of an ANN, as shown in figure 17. Another approach is to cluster features, e.g. by tSNE [1584, 1585] with the Barnes–Hut algorithm [1586, 1587], and examine corresponding clustering of inputs or outputs [231]. Finally, developers can view raw features and gradients during forward and backward passes of gradient descent, respectively. For example, CNN explainer

Figure 17. Inputs that maximally activate channels in GoogLeNet [1076] after training on ImageNet [71]. Neurons in layers near the start have small receptive fields and discern local features. Middle layers discern semantics recognisable by humans, such as dogs and wheels. Finally, layers at the end of the DNN, near its logits, discern combinations of semantics that are useful for labelling. This figure is adapted with permission [1553] under a Creative Commons Attribution 4.0 [73] license.

[1588, 1589] is an interactive visualization tool designed for non-experts to learn and experiment with CNNs. Similarly, GAN Lab [1590] is an interactive visualization tool for non-experts to learn and experiment with GANs.

7. Discussion

We introduced a variety of electron microscopy applications in section 1 that have been enabled or enhanced by deep learning. Nevertheless, the greatest benefit of deep learning in electron microscopy may be general-purpose tools that enable researchers to be more effective. Search engines based on deep learning are almost essential to navigate an ever-increasing number of scientific publications [700]. Further, machine learning can enhance communication by filtering spam and phishing attacks [1591–1593], and by summarizing [1594–1596] and classifying [1055, 1597–1599] scientific documents. In addition, machine learning can be applied to education to automate and standardize scoring [1600–1603], detect plagiarism [1604–1606], and identify at-risk students [1607].

Creative applications of deep learning [1608, 1609] include making new art by style transfer [1001–1005], composing music [1610–1612], and storytelling [1613, 1614]. Similar DNNs can assist programmers [1615, 1616]. For example, by predictive source code completion [1617–1622], and by generating source code to map inputs to target outputs [1623] or from labels describing desired source code [1624]. Text generating DNNs can also help write scientific papers. For example, by drafting scientific passages [1625] or drafting part of a paper from a list of references [1626]. Papers generated by early prototypes for automatic scientific paper generators, such as SciGen [1627], are realistic insofar that they have been accepted by scientific venues.

An emerging application of deep learning is mining scientific resources to make new scientific discoveries [1628]. Artificial agents are able to effectively distil latent scientific knowledge as they can parallelize examination of huge amounts of data, whereas information access by humans [1629–1631] is limited by human cognition [1632]. High bandwidth bi-directional brain-machine interfaces are being developed to overcome limitations of human cognition [1633]; however, they are in the early stages of development and we expect that they will depend on substantial advances in machine learning to enhance control of cognition. Eventually, we expect that ANNs will be used as scientific oracles, where researchers who do not rely on their services will no longer be able to compete. For example, an ANN trained on a large corpus of scientific literature predicted multiple advances in materials science before they were reported [1634]. ANNs are

already used for financial asset management [1635, 1636] and recruiting [1637–1640], so we anticipate that artificial scientific oracle consultation will become an important part of scientific grant [1641, 1642] reviews.

A limitation of deep learning is that it can introduce new issues. For example, DNNs are often susceptible to adversarial attacks [1643–1647], where small perturbations to inputs cause large errors. Nevertheless, training can be modified to improve robustness to adversarial attacks [1648–1652]. Another potential issue is architecture-specific systematic errors. For example, CNNs often exhibit structured systematic error variation [70, 201, 202, 1092, 1093, 1653], including higher errors nearer output edges [70, 201, 202]. However, structured systematic error variation can be minimized by GANs incentifying the generation of realistic outputs [201]. Finally, ANNs can be difficult to use as they often require downloading code with undocumented dependencies, downloading a pretrained model, and may require hardware accelerators. These issues can be avoided by serving ANNs from cloud containers. However, it may not be practical for academics to acquire funding to cover cloud service costs.

Perhaps the most important aspect of deep learning in electron microscopy is that it presents new challenges that can lead to advances in machine learning. Simple benchmarks like CIFAR-10 [562, 563] and MNIST [564] have been solved. Following, more difficult benchmarks like Fashion-MNIST [1654] have been introduced. However, they only partially address issues with solved datasets as they do not present fundamentally new challenges. In contrast, we believe that new problems often invite new solutions. For example, we developed adaptive learning rate clipping [261] to stabilize training of DNNs for partial STEM [201]. The challenge was that we wanted to train a large model for high-resolution images; however, training was unstable if we used small batches needed to fit it in GPU memory. Similar challenges abound and can lead to advances in both machine learning and electron microscopy.

Data availability statement

No new data were created or analysed in this study.

Acknowledgments

Thanks go to Jeremy Sloan and Martin Lotz for internally reviewing this article. In addition, part of the text in section 1.2 is adapted from our earlier work with permission [201] under a Creative Commons Attribution 4.0 [73] license. Finally, the author acknowledges funding from EPSRC grant EP/N035437/1 and EPSRC Studentship 1917382.

Conflict of interest

The author declares no competing interests.

ORCID iD

Jeffrey M Ede https://orcid.org/0000-0002-9358-5364

References

[1] Leiserson C E *et al* 2020 There's plenty of room at the top: what will drive computer performance after Moore's law? *Science* **368**
[2] Sun C, Shrivastava A, Singh S and Gupta A 2017 Revisiting unreasonable effectiveness of data in deep learning era *Proc. IEEE Int. Conf. on Computer Vision* pp 843–52
[3] Hey T, Butler K, Jackson S and Thiyagalingam J 2020 Machine learning and big scientific data *Phil. Trans. R. Soc.* A **378** 20190054
[4] Sengupta S *et al* 2020 A review of deep learning with special emphasis on architectures, applications and recent trends *Knowl.-Based Syst.* **4** 105596
[5] Shrestha A and Mahmood A 2019 Review of deep learning algorithms and architectures *IEEE Access* **7** 53040–65
[6] Dargan S, Kumar M, Ayyagari M R and Kumar G 2019 A survey of deep learning and its applications: a new paradigm to machine learning *Archives Computat. Methods Eng.* **2** 1071–92
[7] Alom M Z *et al* 2019 A state-of-the-art survey on deep learning theory and architectures *Electronics* **8** 292
[8] Zhang Q, Yang L T, Chen Z and Li P 2018 A survey on deep learning for big data *Inform. Fusion* **42** 146–57
[9] Hatcher W G and Yu W 2018 A survey of deep learning: platforms, applications and emerging research trends *IEEE Access* **6** 24411–32
[10] LeCun Y, Bengio Y and Hinton G 2015 Deep learning *Nature* **521** 436–44
[11] Schmidhuber J 2015 Deep learning in neural networks: an overview *Neural Netw.* **61** 85–117
[12] Ge M, Su F, Zhao Z and Su D 2020 Deep learning analysis on microscopic imaging in materials science *Mater. Today Nano* **11** 100087
[13] Carleo G *et al* 2019 Machine learning and the physical sciences *Rev. Mod. Phys.* **91** 045002
[14] Wei J *et al* 2019 Machine learning in materials science *InfoMat* **1** 338–58
[15] Barbastathis G, Ozcan A and Situ G 2019 On the use of deep learning for computational imaging *Optica* **6** 921–43

[16] Schleder G R, Padilha A C, Acosta C M, Costa M and Fazzio A 2019 From DFT to machine learning: recent approaches to materials science—a review *J. Phys.: Mater.* **2** 032001
[17] von Lilienfeld O A 2020 Introducing machine learning: science and technology *Mach. Learn.: Sci. Technol.* **1** 010201
[18] Sejnowski T J 2018 *The Deep Learning Revolution* (Cambridge, MA: MIT Press)
[19] Alom M Z *et al* 2018 The history began from AlexNet: a comprehensive survey on deep learning approaches (arXiv:1803.01164)
[20] Wang Y and Kosinski M 2018 Deep neural networks are more accurate than humans at detecting sexual orientation from facial images *J. Pers. Soc. Psychol.* **114** 246
[21] Kheradpisheh S R, Ghodrati M, Ganjtabesh M and Masquelier T 2016 Deep networks can resemble human feed-forward vision in invariant object recognition *Sci. Rep.* **6** 32672
[22] He K, Zhang X, Ren S and Sun J 2015 Delving deep into rectifiers: surpassing human-level performance on imagenet classification *Proc. IEEE Int. Conf. on Computer Vision* pp 1026–34
[23] Lu C and Tang X 2015 Surpassing human-level face verification performance on LFW with gaussianface *Twenty-Ninth Conf. on Artificial Intelligence*
[24] Vinyals O *et al* 2019 AlphaStar: mastering the real-time strategy game StarCraft II (available online at: https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/)
[25] Firoiu V, Whitney W F and Tenenbaum J B 2017 Beating the world's best at super smash bros. with deep reinforcement learning (arXiv:1702.06230)
[26] Lample G and Chaplot D S 2017 Playing FPS games with deep reinforcement learning *31st Conf. on Artificial Intelligence*
[27] Silver D *et al* 2016 Mastering the game of Go with deep neural networks and tree search *Nature* **529** 484–9
[28] Mnih V *et al* 2013 Playing Atari with deep reinforcement learning (arXiv:1312.5602)
[29] Tesauro G 2002 Programming backgammon using self-teaching neural nets *Artif. Intell.* **134** 181–99
[30] Han S S *et al* 2018 Deep neural networks show an equivalent and often superior performance to dermatologists in onychomycosis diagnosis: automatic construction of onychomycosis datasets by region-based convolutional deep neural network *PLoS One* **13** e0191493
[31] Wang D, Khosla A, Gargeya R, Irshad H and Beck A H 2016 Deep learning for identifying metastatic breast cancer (arXiv:1606.05718)
[32] Santoro A *et al* 2017 A simple neural network module for relational reasoning *Adv. Neural Inf. Process. Syst.* 4967–76
[33] Xiong W *et al* 2016 Achieving human parity in conversational speech recognition (arXiv:1610.05256)
[34] Weng C, Yu D, Seltzer M L and Droppo J 2014 Single-channel mixed speech recognition using deep neural networks *2014 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 5632–6
[35] Lee K, Zung J, Li P, Jain V and Seung H S 2017 Superhuman accuracy on the SNEMI3D connectomics challenge (arXiv:1706.00120)
[36] Weyand T, Kostrikov I and Philbin J 2016 Planet-photo geolocation with convolutional neural networks *Conf. on Computer Vision* (Springer) pp 37–55
[37] Kidger P and Lyons T 2019 Universal approximation with deep narrow networks (arXiv:1905.08539)
[38] Lin H and Jegelka S 2018 ResNet with one-neuron hidden layers is a universal approximator *Advances in Neural Information Processing Systems* pp 6169–78
[39] Hanin B and Sellke M 2017 Approximating continuous functions by ReLU nets of minimal width (arXiv:1710.11278)
[40] Lu Z, Pu H, Wang F, Hu Z and Wang L 2017 The expressive power of neural networks: a view from the width *Advances in Neural Information Processing Systems* pp 6231–9
[41] Pinkus A 1999 Approximation theory of the MLP model in neural networks *Acta Numer.* **8** 143–95
[42] Leshno M, Lin V Y, Pinkus A and Schocken S 1993 Multilayer feedforward networks with a nonpolynomial activation function can approximate any function *Neural Netw.* **6** 861–7
[43] Hornik K 1991 Approximation capabilities of multilayer feedforward networks *Neural Netw.* **4** 251–7
[44] Hornik K, Stinchcombe M and White H 1989 Multilayer feedforward networks are universal approximators *Neural Netw.* **2** 359–66
[45] Cybenko G 1989 Approximation by superpositions of a sigmoidal function *Math. Control Signals Syst.* **2** 303–14
[46] Johnson J 2018 Deep, skinny neural networks are not universal approximators (arXiv:1810.00393)
[47] Lin H W, Tegmark M and Rolnick D 2017 Why does deep and cheap learning work so well? *J. Stat. Phys.* **168** 1223–47
[48] Gühring I, Raslan M and Kutyniok G 2020 Expressivity of deep neural networks (arXiv:2007.04759)
[49] Raghu M, Poole B, Kleinberg J, Ganguli S and Sohl-Dickstein J 2017 On the expressive power of deep neural networks *Int. Conf. on Machine Learning* pp 2847–54
[50] Poole B, Lahiri S, Raghu M, Sohl-Dickstein J and Ganguli S 2016 Exponential expressivity in deep neural networks through transient chaos *Advances in Neural Information Processing Systems* pp 3360–8
[51] Hanin B and Rolnick D 2019 Deep ReLU networks have surprisingly few activation patterns *Advances in Neural Information Processing Systems* pp 361–70
[52] Cao Y and Gu Q 2020 Generalization error bounds of gradient descent for learning over-parameterized deep ReLU networks *34th Conf. on Artificial Intelligence* pp 3349–56
[53] Geiger M *et al* 2020 Scaling description of generalization with number of parameters in deep learning *J. Stat. Mech.: Theory Exp.* **2020** 023401
[54] Dziugaite G K 2020 Revisiting generalization for deep learning: PAC-Bayes, flat minima, and generative models PhD Thesis University of Cambridge
[55] Cao Y and Gu Q 2019 Generalization bounds of stochastic gradient descent for wide and deep neural networks *Advances in Neural Information Processing Systems* pp 10836–46
[56] Xu Z J 2018 Understanding training and generalization in deep learning by Fourier analysis (arXiv:1808.04295)
[57] Neyshabur B, Bhojanapalli S, McAllester D and Srebro N 2017 Exploring generalization in deep learning *Advances in Neural Information Processing Systems* pp 5947–56
[58] Wu L, Zhu Z *et al* 2017 Towards understanding generalization of deep learning: perspective of loss landscapes (arXiv:1706.10239)
[59] Kawaguchi K, Kaelbling L P and Bengio Y 2017 Generalization in deep learning (arXiv:1710.05468)
[60] Iten R, Metger T, Wilming H, Del Rio L and Renner R 2020 Discovering physical concepts with neural networks *Phys. Rev. Lett.* **124** 010508
[61] Wu T and Tegmark M 2019 Toward an artificial intelligence physicist for unsupervised learning *Phys. Rev.* E **100** 033311

[62] Chen Y, Xie Y, Song L, Chen F and Tang T 2020 A survey of accelerator architectures for deep neural networks *Engineering* **6** 264–74
[63] Garrido M, Qureshi F, Takala J and Gustafsson O 2019 Hardware architectures for the fast fourier transform *Handbook of Signal Processing Systems* (Springer) pp 613–47
[64] Velik R 2008 Discrete fourier transform computation using neural networks *2008 Int. Conf. on Computational Intelligence and Security* (IEEE) pp 120–3
[65] Moreland K and Angel E 2003 The FFT on a GPU *Proc. ACM SIGGRAPH/ Conf. on Graphics Hardware* (Eurographics Association) pp 112–19
[66] Breen P G, Foley C N, Boekholt T and Zwart S P 2020 Newton versus the machine: solving the chaotic three-body problem using deep neural networks *Mon. Not. R. Astron. Soc.* **494** 2465–70
[67] Ryczko K, Strubbe D A and Tamblyn I 2019 Deep learning and density-functional theory *Phys. Rev.* A **100** 022512
[68] Sinitskiy A V and Pande V S 2018 Deep neural network computes electron densities and energies of a large set of organic molecules faster than density functional theory (DFT) (arXiv:1809.02723)
[69] Zhang G *et al* 2018 Fast phase retrieval in off-axis digital holographic microscopy through deep learning *Opt. Express* **26** 19388–405
[70] Ede J M and Beanland R 2019 Improving electron micrograph signal-to-noise with an atrous convolutional encoder–decoder *Ultramicroscopy* **202** 18–25
[71] Krizhevsky A, Sutskever I and Hinton G E 2012 ImageNet classification with deep convolutional neural networks *Advances in Neural Information Processing Systems* pp 1097–105
[72] Ede J M 2018 Improving electron micrograph signal-to-noise with an atrous convolutional encoder–decoder (arXiv:1807.11234)
[73] Creative Commons Attribution 4.0 Int. (CC BY 4.0) 2020 (available online at: https://creativecommons.org/licenses/by/4.0)
[74] Liu B and Liu J 2019 Overview of image denoising based on deep learning *J. Phys.: Conf. Ser.* **1176** 022010
[75] Tian C *et al* 2019 Deep learning on image denoising: an overview (arXiv:1912.13171)
[76] Yoon D, Lim H S, Jung K, Kim T Y and Lee S 2019 Deep learning-based electrocardiogram signal noise detection and screening model *Healthc. Inform. Res.* **25** 201–11
[77] Antczak K 2018 Deep recurrent neural networks for ECG signal denoising (arXiv:1807.11551)
[78] Bai T, Nguyen D, Wang B and Jiang S 2020 Probabilistic self-learning framework for low-dose CT denoising (arXiv:2006.00327)
[79] Jifara W, Jiang F, Rho S, Cheng M and Liu S 2019 Medical image denoising using convolutional neural network: a residual learning approach *J. Supercomput.* **75** 704–18
[80] Feng D, Wu W, Li H and Li Q 2019 Speckle noise removal in ultrasound images using a deep convolutional neural network and a specially designed loss function *Int. Workshop on Multiscale Multimodal Medical Imaging* (Springer) pp 85–92
[81] de Haan K, Rivenson Y, Wu Y and Ozcan A 2019 Deep-learning-based image reconstruction and enhancement in optical microscopy *Proc. IEEE* **108** 30–50
[82] Manifold B, Thomas E, Francis A T, Hill A H and Fu D 2019 Denoising of stimulated raman scattering microscopy images via deep learning *Biomed. Opt. Express* **10** 3860–74
[83] Devalla S K *et al* 2019 A deep learning approach to denoise optical coherence tomography images of the optic nerve head *Sci. Rep.* **9** 1–13
[84] Choi G *et al* 2019 Cycle-consistent deep learning approach to coherent noise reduction in optical diffraction tomography *Opt. Express* **27** 4927–43
[85] Azarang A and Kehtarnavaz N 2020 A review of multi-objective deep learning speech denoising methods *Speech Commun.*
[86] Choi H-S, Heo H, Lee J H and Lee K 2020 Phase-aware single-stage speech denoising and dereverberation with U-net (arXiv:2006.00687)
[87] Alamdari N, Azarang A and Kehtarnavaz N 2019 Self-supervised deep learning-based speech denoising (arXiv:1904)
[88] Han K *et al* 2015 Learning spectral mapping for speech dereverberation and denoising *IEEE/ACM Trans. Audio Speech Lang. Process.* **23** 982–92
[89] Goyal B, Dogra A, Agrawal S, Sohi B and Sharma A 2020 Image denoising review: from classical to state-of-the-art approaches *Inf. Fusion* **55** 220–44
[90] Girdher A, Goyal B, Dogra A, Dhindsa A and Agrawal S 2019 Image denoising: issues and challenges *Available at SSRN 3446627*
[91] Fan L, Zhang F, Fan H and Zhang C 2019 Brief review of image denoising techniques *Vis. Comput. Ind. Biomed. Art* **2** 7
[92] Gedraite E S and Hadad M 2011 Investigation on the effect of a gaussian blur in image filtering and segmentation *Proc. ELMAR* (IEEE) pp 393–6
[93] Deng G and Cahill L 1993 An adaptive gaussian filter for noise reduction and edge detection *1993 Conf. Record Nuclear Symp. and Medical Conf.* (IEEE) pp 1615–19
[94] Chang H-H, Lin Y-J and Zhuang A H 2019 An automatic parameter decision system of bilateral filtering with GPU-based acceleration for brain MR images *J. Digit. Imaging* **32** 148–61
[95] Chaudhury K N and Rithwik K 2015 Image denoising using optimally weighted bilateral filters: a sure and fast approach *IEEE Int. Conf. on Image Processing* (IEEE) pp 108–12
[96] Anantrasirichai N *et al* 2014 Adaptive-weighted bilateral filtering and other pre-processing techniques for optical coherence tomography *Comput. Med. Imaging Graph.* **38** 526–39
[97] Tomasi C and Manduchi R 1998 Bilateral filtering for gray and color images *6th Int. Conf. on Computer Vision (IEEE Cat. No. 98CH36271)* (IEEE) pp 839–46
[98] Budhiraja S, Goyal B, Dogra A and Agrawal S *et al* 2018 An efficient image denoising scheme for higher noise levels using spatial domain filters *Biomed. Pharmacol. J.* **11** 625–34
[99] Nair R R, David E and Rajagopal S 2019 A robust anisotropic diffusion filter with low arithmetic complexity for images *EURASIP J. Image Video Process.* **2019** 48
[100] Perona P and Malik J 1990 Scale-space and edge detection using anisotropic diffusion *IEEE Trans. Pattern Anal. Mach. Intell.* **12** 629–39
[101] Wang Z and Zhang D 1999 Progressive switching median filter for the removal of impulse noise from highly corrupted images *IEEE Trans. Circuits Syst.* **46** 78–80
[102] Yang R, Yin L, Gabbouj M, Astola J and Neuvo Y 1995 Optimal weighted median filtering under structural constraints *IEEE Trans. Signal Process.* **43** 591–604
[103] Kodi Ramanah D, Lavaux G and Wandelt B D 2017 Wiener filter reloaded: fast signal reconstruction without preconditioning *Mon. Not. R. Astron. Soc.* **468** 1782–93

[104] Elsner F and Wandelt B D 2013 Efficient wiener filtering without preconditioning *Astron. Astrophys.* **549** A111
[105] Robinson E A and Treitel S 1967 Principles of digital wiener filtering *Geophys. Prospect.* **15** 311–32
[106] Bayer F M, Kozakevicius A J and Cintra R J 2019 An iterative wavelet threshold for signal denoising *Signal Process.* **162** 10–20
[107] Mohideen S K, Perumal S A and Sathik M M 2008 Image de-noising using discrete wavelet transform *Int. J. Comput. Sci. Netw. Secur.* **8** 213–16
[108] Luisier F, Blu T and Unser M 2007 A new sure approach to image denoising: interscale orthonormal wavelet thresholding *IEEE Trans. Image Process.* **16** 593–606
[109] Jansen M and Bultheel A 2001 Empirical bayes approach to improve wavelet thresholding for image noise reduction *J. Am. Stat. Assoc.* **96** 629–39
[110] Chang S G, Yu B and Vetterli M 2000 Adaptive wavelet thresholding for image denoising and compression *IEEE Trans. Image Process.* **9** 1532–46
[111] Donoho D L and Johnstone J M 1994 Ideal spatial adaptation by wavelet shrinkage *Biometrika* **81** 425–55
[112] Ma J and Plonka G 2010 The curvelet transform *IEEE Signal Process. Mag.* **27** 118–33
[113] Starck J-L, Candès E J and Donoho D L 2002 The curvelet transform for image denoising *IEEE Trans. Image Process.* **11** 670–84
[114] Ahmed S S *et al* 2015 Nonparametric denoising methods based on contourlet transform with sharp frequency localization: application to low exposure time electron microscopy images *Entropy* **17** 3461–78
[115] Do M N and Vetterli M 2005 The contourlet transform: an efficient directional multiresolution image representation *IEEE Trans. Image Process.* **14** 2091–106
[116] Diwakar M and Kumar P 2019 Wavelet packet based ct image denoising using bilateral method and bayes shrinkage rule *Handbook of Multimedia Information Security: Techniques and Applications* (Springer) pp 501–11
[117] Thakur K, Damodare O and Sapkal A 2015 Hybrid method for medical image denoising using shearlet transform and bilateral filter *2015 Int. Conf. on Information Processing (ICIP)* (IEEE) pp 220–4
[118] Nagu M and Shanker N V 2014 Image de-noising by using median filter and weiner filter *Image* **2** 5641–9
[119] Bae T-W 2014 Spatial and temporal bilateral filter for infrared small target enhancement *Infrared Phys. Technol.* **63** 42–53
[120] Knaus C and Zwicker M 2013 Dual-domain image denoising *2013 IEEE Int. Conf. on Image Processing* (IEEE) pp 440–4
[121] Danielyan A, Katkovnik V and Egiazarian K 2011 BM3D frames and variational image deblurring *IEEE Trans. Image Process.* **21** 1715–28
[122] Dabov K, Foi A, Katkovnik V and Egiazarian K 2007 Image denoising by sparse 3-D transform-domain collaborative filtering *IEEE Trans. Image Process.* **16** 2080–95
[123] Jia L *et al* 2017 Image denoising via sparse representation over grouped dictionaries with adaptive atom size *IEEE Access* **5** 22514–29
[124] Shao L, Yan R, Li X and Liu Y 2013 From heuristic optimization to dictionary learning: a review and comprehensive comparison of image denoising algorithms *IEEE Trans. Cybern.* **44** 1001–13
[125] Chatterjee P and Milanfar P 2009 Clustering-based denoising with locally learned dictionaries *IEEE Trans. Image Process.* **18** 1438–51
[126] Aharon M, Elad M and Bruckstein A 2006 K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation *IEEE Trans. Signal Process.* **54** 4311–22
[127] Elad M and Aharon M 2006 Image denoising via sparse and redundant representations over learned dictionaries *IEEE Trans. Image Process.* **15** 3736–45
[128] Pairis S *et al* 2019 Shot-noise-limited nanomechanical detection and radiation pressure backaction from an electron beam *Phys. Rev. Lett.* **122** 083603
[129] Seki T, Ikuhara Y and Shibata N 2018 Theoretical framework of statistical noise in scanning transmission electron microscopy *Ultramicroscopy* **193** 118–25
[130] Lee Z, Rose H, Lehtinen O, Biskupek J and Kaiser U 2014 Electron dose dependence of signal-to-noise ratio, atom contrast and resolution in transmission electron microscope images *Ultramicroscopy* **145** 3–12
[131] Timischl F, Date M and Nemoto S 2012 A statistical model of signal–noise in scanning electron microscopy *Scanning* **34** 137–44
[132] Sim K, Thong J and Phang J 2004 Effect of shot noise and secondary emission noise in scanning electron microscope images *Scanning* **26** 36–40
[133] Boyat A K and Joshi B K 2015 A review paper: noise models in digital image processing (arXiv:1505.03489)
[134] Meyer R R and Kirkland A I 2000 Characterisation of the signal and noise transfer of CCD cameras for electron detection *Microsc. Res. Tech.* **49** 269–80
[135] Kujawa S and Krahl D 1992 Performance of a low-noise ccd camera adapted to a transmission electron microscope *Ultramicroscopy* **46** 395–403
[136] Rose H H 2008 Optics of high-performance electron microscopes *Sci. Technol. Adv. Mater.* **9** 014107
[137] Fujinaka S, Sato Y, Teranishi R and Kaneko K 2020 Understanding of scanning-system distortions of atomic-scale scanning transmission electron microscopy images for accurate lattice parameter measurements *J. Mater. Sci.* **55** 8123–33
[138] Sang X *et al* 2016 Dynamic scan control in stem: spiral scans *Adv. Struct. Chem. Imaging* **2** 1–8
[139] Ning S *et al* 2018 Scanning distortion correction in stem images *Ultramicroscopy* **184** 274–83
[140] Ophus C, Ciston J and Nelson C T 2016 Correcting nonlinear drift distortion of scanning probe and scanning transmission electron microscopies from image pairs with orthogonal scan directions *Ultramicroscopy* **162** 1–9
[141] Jones L and Nellist P D 2013 Identifying and correcting scan noise and drift in the scanning transmission electron microscope *Microsc. Microanal.* **19** 1050–60
[142] Karthik C, Kane J, Butt D P, Windes W and Ubic R 2011 *In situ* transmission electron microscopy of electron-beam induced damage process in nuclear grade graphite *J. Nucl. Mater.* **412** 321–6
[143] Roels J *et al* 2020 An interactive ImageJ plugin for semi-automated image denoising in electron microscopy *Nat. Commun.* **11** 1–13
[144] Narasimha R *et al* 2008 Evaluation of denoising algorithms for biological electron tomography *J. Struct. Biol.* **164** 7–17
[145] Mevenkamp N *et al* 2015 Poisson noise removal from high-resolution stem images based on periodic block matching *Adv. Struct. Chem. Imaging* **1** 3
[146] Bajić B, Lindblad J and Sladoje N 2016 Blind restoration of images degraded with mixed poisson-gaussian noise with application in transmission electron microscopy *2016 IEEE 13th Int. Symp. on Biomedical Imaging (ISBI)* (IEEE) pp 123–7
[147] Bodduna K and Weickert J 2020 Image denoising with less artefacts: novel non-linear filtering on fast patch reorderings (arXiv:2002.00638)

[148] Jonić S *et al* 2016 Denoising of high-resolution single-particle electron-microscopy density maps by their approximation using three-dimensional gaussian functions *J. Struct. Biol.* **194** 423–33
[149] Chung S-C *et al* 2020 Two-stage dimension reduction for noisy high-dimensional images and application to cryogenic electron microscopy (arXiv::1911)
[150] Wang J and Yin C 2013 A Zernike-moment-based non-local denoising filter for cryo-em images *Sci. China Life Sci.* **56** 384–90
[151] Furnival T, Leary R K and Midgley P A 2017 Denoising time-resolved microscopy image sequences with singular value thresholding *Ultramicroscopy* **178** 112–24
[152] Sorzano C O S, Ortiz E, López M and Rodrigo J 2006 Improved Bayesian image denoising based on wavelets with applications to electron microscopy *Pattern Recognit.* **39** 1205–13
[153] Ouyang J *et al* 2018 Cryo-electron microscope image denoising based on the geodesic distance *BMC Struct. Biol.* **18** 18
[154] Du H 2015 A nonlinear filtering algorithm for denoising HR (S)TEM micrographs *Ultramicroscopy* **151** 62–7
[155] Kushwaha H S, Tanwar S, Rathore K and Srivastava S 2012 De-noising filters for TEM (transmission electron microscopy) image of nanomaterials *2012 2nd Int. Conf. on Advanced Computing and Communication Technologies* (IEEE) pp 276–81
[156] Hanai T, Morinaga T, Suzuki H and Hibino M 1997 Maximum entropy restoration of electron microscope images with a random-spatial-distribution constraint *Scanning Microsc.* **11** 379–90
[157] Pennycook S J 2017 The impact of stem aberration correction on materials science *Ultramicroscopy* **180** 22–33
[158] Ramasse Q M 2017 Twenty years after: how 'aberration correction in the stem' truly placed a 'a synchrotron in a microscope' *Ultramicroscopy* **180** 41–51
[159] Hawkes P 2009 Aberration correction past and present *Philos. Trans. R. Soc.* A **367** 3637–64
[160] Goodge B H, Bianco E and Kourkoutis H W 2020 Atomic-resolution cryo-stem across continuously variable temperature (arXiv:2001.11581)
[161] Egerton R F 2019 Radiation damage to organic and inorganic specimens in the TEM *Micron* **119** 72–87
[162] Egerton R F 2013 Control of radiation damage in the TEM *Ultramicroscopy* **127** 100–8
[163] Egerton R 2012 Mechanisms of radiation damage in beam-sensitive specimens, for tem accelerating voltages between 10 and 300 kV *Microsc. Res. Tech.* **75** 1550–6
[164] Mankos M *et al* 2019 Electron optics for a multi-pass transmission electron microscope *Adv. Imaging Electron Phys.* **212** 71–86
[165] Koppell S A *et al* 2019 Design for a 10 keV multi-pass transmission electron microscope *Ultramicroscopy* **207** 112834
[166] Juffmann T *et al* 2017 Multi-pass transmission electron microscopy *Sci. Rep.* **7** 1–7
[167] Jones L *et al* 2018 Managing dose-, damage- and data-rates in multi-frame spectrum-imaging *Microscopy* **67** i98–i113
[168] Krull A, Buchholz T-O and Jug F 2019 Noise2Void—learning denoising from single noisy images *Proc. Conf. on Computer Vision and Pattern Recognition* pp 2129–37
[169] Guo S, Yan Z, Zhang K, Zuo W and Zhang L 2019 Toward convolutional blind denoising of real photographs *Proc. Conf. on Computer Vision and Pattern Recognition* pp 1712–22
[170] Lefkimmiatis S 2018 Universal denoising networks: a novel cnn architecture for image denoising *Proc. Conf. on Computer Vision and Pattern Recognition* pp 3204–13
[171] Weigert M *et al* 2018 Content-aware image restoration: pushing the limits of fluorescence microscopy *Nat. Methods* **15** 1090–7
[172] Zhang K, Zuo W and Zhang L 2018 FFDNet: toward a fast and flexible solution for cnn-based image denoising *IEEE Trans. Image Process.* **27** 4608–22
[173] Weigert M, Royer L, Jug F and Myers G 2017 Isotropic reconstruction of 3D fluorescence microscopy images using convolutional neural networks *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention* (Springer) pp 126–34
[174] Zhang K, Zuo W, Chen Y, Meng D and Zhang L 2017 Beyond a Gaussian denoiser: residual learning of deep CNN for image denoising *IEEE Trans. Image Process.* **26** 3142–55
[175] Tai Y, Yang J, Liu X and Xu C 2017 MemNet: a persistent memory network for image restoration *Proc. IEEE Int. Conf. on Computer Vision* pp 4539–47
[176] Mao X, Shen C and Yang Y-B 2016 Image restoration using very deep convolutional encoder–decoder networks with symmetric skip connections *Advances in Neural Information Processing Systems* pp 2802–10
[177] Buchholz T-O, Jordan M, Pigino G and Jug F 2019 Cryo-CARE: content-aware image restoration for cryo-transmission electron microscopy data *2019 IEEE 16th Int. Symp. on Biomedical Imaging (ISBI 2019)* (IEEE) pp 502–6
[178] Fang L *et al* 2019 Deep learning-based point-scanning super-resolution imaging *bioRxiv* 740548
[179] Mohan S *et al* 2020 Deep denoising for scientific discovery: a case study in electron microscopy (arXiv:2010.12970)
[180] Giannatou E, Papavieros G, Constantoudis V, Papageorgiou H and Gogolides E 2019 Deep learning denoising of sem images towards noise-reduced ler measurements *Microelectron. Eng.* **216** 111051
[181] Chaudhary N, Savari S A and Yeddulapalli S S 2019 Line roughness estimation and poisson denoising in scanning electron microscope images using deep learning *J. Micro. Nanolithogr. MEMS MOEMS* **18** 024001
[182] Vasudevan R K and Jesse S 2019 Deep learning as a tool for image denoising and drift correction *Microsc. Microanal.* **25** 190–1
[183] Wang F, Henninen T R, Keller D and Erni R 2020 Noise2Atom: unsupervised denoising for scanning transmission electron microscopy images *Res. Square*
[184] Bepler T, Noble A J and Berger B 2019 Topaz-denoise: general deep denoising models for cryoEM *bioRxiv* 838920
[185] Lehtinen J *et al* 2018 Noise2Noise: learning image restoration without clean data *Int. Conf. on Machine Learning* pp 2965–74
[186] Tegunov D and Cramer P 2019 Real-time cryo-electron microscopy data preprocessing with warp *Nat. Methods* **16** 1146–52
[187] Zhang C, Berkels B, Wirth B and Voyles P M 2017 Joint denoising and distortion correction for atomic column detection in scanning transmission electron microscopy images *Microsc. Microanal.* **23** 164–5
[188] Jin P and Li X 2015 Correction of image drift and distortion in a scanning electron microscopy *J. Microsc.* **260** 268–80
[189] Tong X *et al* 2019 Image registration with Fourier-based image correlation: a comprehensive review of developments and applications *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **12** 4062–81
[190] Krishnan A P *et al* 2020 Optical aberration correction via phase diversity and deep learning *bioRxiv*
[191] Cumming B P and Gu M 2020 Direct determination of aberration functions in microscopy by an artificial neural network *Opt. Express* **28** 14511–21
[192] Wang W, Wu B, Zhang B, Li X and Tan J 2020 Correction of refractive index mismatch-induced aberrations under radially polarized illumination by deep learning *Opt. Express* **28** 26028–40
[193] Tian Q *et al* 2019 DNN-Based aberration correction in a wavefront sensorless adaptive optics system *Opt. Express* **27** 10765–76
[194] Rivenson Y *et al* 2018 Deep learning enhanced mobile-phone microscopy *ACS Photonics* **5** 2354–64

[195] Nguyen T *et al* 2017 Automatic phase aberration compensation for digital holographic microscopy based on deep learning background detection *Opt. Express* **25** 15043–57
[196] Jeon S and Kim C 2020 Deep learning-based speed of sound aberration correction in photoacoustic images *Photons Plus Ultrasound: Imaging and Sensing 2020* vol 11240 (Int. Society for Optics and Photonics) p 112400J
[197] Gui J, Sun Z, Wen Y, Tao D and Ye J 2020 A review on generative adversarial networks: algorithms, theory, and applications (arXiv:2001.06937)
[198] Saxena D and Cao J 2020 Generative adversarial networks (GANs): challenges, solutions, and future directions (arXiv:2005.00065)
[199] Pan Z *et al* 2019 Recent progress on generative adversarial networks (GANs): a survey *IEEE Access* **7** 36322–33
[200] Wang Z, She Q and Ward T E 2019 Generative adversarial networks: a survey and taxonomy (arXiv:1906.01529)
[201] Ede J M and Beanland R 2020 Partial scanning transmission electron microscopy with deep learning *Sci. Rep.* **10** 1–10
[202] Ede J M 2019 Deep learning supersampled scanning transmission electron microscopy (arXiv:1910.10467)
[203] Atta R E, Kasem H M and Attia M 2020 A Comparison study for image compression based on compressive sensing *Eleventh Int. Conf. on Graphics and Image Processing (ICGIP 2019)* vol 11373 (Int. Society for Optics and Photonics) p 1137315
[204] Vidyasagar M 2019 *An Introduction to Compressed Sensing* (Philadelphia, PA: SIAM)
[205] Rani M, Dhok S B and Deshmukh R 2018 A systematic review of compressive sensing: concepts, implementations and applications *IEEE Access* **6** 4875–94
[206] Eldar Y C and Kutyniok G 2012 *Compressed Sensing: Theory and Applications* (Cambridge: Cambridge University Press)
[207] Donoho D L 2006 Compressed sensing *IEEE Trans. Inf. Theory* **52** 1289–306
[208] Johnson P M, Recht M P and Knoll F 2020 Improving the speed of MRI with artificial intelligence *Seminars in Musculoskeletal Radiology* vol 24 (NIH Public Access) p 12
[209] Ye J C 2019 Compressed sensing MRI: a review from signal processing perspective *BMC Biomed. Eng.* **1** 1–17
[210] Lustig M, Donoho D and Pauly J M 2007 Sparse MRI: the application of compressed sensing for rapid MR imaging *Magn. Reson. Med.* **58** 1182–95
[211] Yuan X and Haimi-Cohen R 2020 Image compression based on compressive sensing: end-to-end comparison with JPEG *IEEE Trans. Multimedia* **22** 2889–904
[212] Gunasheela S and Prasantha H 2019 Compressed sensing for image compression: survey of algorithms *Emerging Research in Computing, Information, Communication and Applications* (Springer) pp 507–17
[213] Wang Z, Chen J and Hoi S C H 2020 Deep learning for image super-resolution: a survey *IEEE Trans. Pattern Anal. Mach. Intell.*
[214] Yang W *et al* 2019 Deep learning for single image super-resolution: a brief review *IEEE Trans. Multimedia* **21** 3106–21
[215] Shin Y J *et al* 2020 Low-Dose abdominal CT using a deep learning-based denoising algorithm: a comparison with CT reconstructed with filtered back projection or iterative reconstruction algorithm *Korean J. Radiol.* **21** 356–64
[216] Cong W *et al* 2019 Deep-learning-based breast ct for radiation dose reduction *Developments in X-ray Tomography XII* vol 11113 (Int. Society for Optics and Photonics) p 111131L
[217] Barkan O, Weill J, Averbuch A and Dekel S 2013 Adaptive compressed tomography sensing *Proc. Conf. on Computer Vision and Pattern Recognition* pp 2195–202
[218] Almasri F and Debeir O 2020 Robust perceptual night vision in thermal colorization (arXiv:2003.02204)
[219] Chen C, Chen Q, Xu J and Koltun V 2018 Learning to see in the dark *Proc. Conf. on Computer Vision and Pattern Recognition* pp 3291–300
[220] Peet M J, Henderson R and Russo C J 2019 The energy dependence of contrast and damage in electron cryomicroscopy of biological molecules *Ultramicroscopy* **203** 125–31
[221] Zhang X *et al* 2018 Radiation damage in nanostructured materials *Prog. Mater. Sci.* **96** 217–321
[222] Lehnert T, Lehtinen O, Algara-Siller G and Kaiser U 2017 Electron radiation damage mechanisms in 2D $MoSe_2$ *Appl. Phys. Lett.* **110** 033106
[223] Hermannsdörfer J, Tinnemann V, Peckys D B and de Jonge N 2016 The effect of electron beam irradiation in environmental scanning transmission electron microscopy of whole cells in liquid *Microsc. Microanal.* **22** 656–65
[224] Johnston-Peck A C, DuChene J S, Roberts A D, Wei W D and Herzing A A 2016 Dose-rate-dependent damage of cerium dioxide in the scanning transmission electron microscope *Ultramicroscopy* **170** 1–9
[225] Jenkins M L and Kirk M A 2000 *Characterisation of Radiation Damage by Transmission Electron Microscopy* (Boca Raton, FL: CRC Press)
[226] Egerton R F, Li P and Malac M 2004 Radiation damage in the TEM and SEM *Micron* **35** 399–409
[227] S'ari M, Cattle J, Hondow N, Brydson R and Brown A 2019 Low dose scanning transmission electron microscopy of organic crystals by scanning moiré fringes *Micron* **120** 1–9
[228] Mayoral A, Mahugo R, Sánchez-Sánchez M and Díaz I 2017 Cs-corrected STEM imaging of both pure and silver-supported metal-organic framework MIL-100 (Fe) *ChemCatChem* **9** 3497–502
[229] Gnanasekaran K, de With G and Friedrich H 2018 Quantification and optimization of ADF-STEM image contrast for beam-sensitive materials *R. Soc. Open Sci.* **5** 171838
[230] Ilett M, Brydson R, Brown A and Hondow N 2019 Cryo-analytical STEM of frozen, aqueous dispersions of nanoparticles *Micron* **120** 35–42
[231] Ede J M 2020 Warwick electron microscopy datasets *Mach. Learn.: Sci. Technol.* **1** 045003
[232] Landau H J 1967 Sampling, data transmission and the nyquist rate *Proc. IEEE* **55** 1701–6
[233] Amidror I 2015 Sub-Nyquist artefacts and sampling moiré effects *R. Soc. Open Sci.* **2** 140550
[234] Fadnavis S 2014 Image interpolation techniques in digital image processing: an overview *Int. J. Eng. Res. Appl.* **4** 70–3
[235] Getreuer P 2011 Linear methods for image interpolation *Image Process. On Line* **1** 238–59
[236] Turkowski K 1990 Filters for common resampling tasks *Graphics Gems* (Morgan Kaufmann) pp 147–65
[237] Beretta L and Santaniello A 2016 Nearest neighbor imputation algorithms: a critical evaluation *BMC Med. Inform. Decis. Mak.* **16** 74
[238] Alfeld P 1984 A trivariate clough–tocher scheme for tetrahedral data *Comput. Aided Geom. Des.* **1** 169–81
[239] Cruz C, Mehta R, Katkovnik V and Egiazarian K O 2017 Single image super-resolution based on wiener filter in similarity domain *IEEE Trans. Image Process.* **27** 1376–89
[240] Zulkifli N, Karim S, Shafie A and Sarfraz M 2019 Rational bicubic ball for image interpolation *J. Phys.: Conf. Series* **1366** 012097
[241] Costella J 2017 The magic kernel Towards Data Science (available at: https://web.archive.org/web/20170707165835/http://johncostella.webs.com/magic)

[242] Olivier R and Hanqiang C 2012 Nearest neighbor value interpolation *Int. J. Adv. Comput. Sci. Appl.* **3** 25–30
[243] Jones L *et al* 2018 Managing dose-, damage- and data-rates in multi-frame spectrum-imaging *Microscopy* **67** i98–113
[244] Trampert P *et al* 2018 How should a fixed budget of dwell time be spent in scanning electron microscopy to optimize image quality? *Ultramicroscopy* **191** 11–7
[245] Stevens A *et al* 2018 A sub-sampled approach to extremely low-dose STEM *Appl. Phys. Lett.* **112** 043104
[246] Hwang S, Han C W, Venkatakrishnan S V, Bouman C A and Ortalan V 2017 Towards the low-dose characterization of beam sensitive nanostructures via implementation of sparse image acquisition in scanning transmission electron microscopy *Meas. Sci. Technol.* **28** 045402
[247] Hujsak K, Myers B D, Roth E, Li Y and Dravid V P 2016 Suppressing electron exposure artifacts: an electron scanning paradigm with bayesian machine learning *Microsc. Microanal.* **22** 778–88
[248] Anderson H S, Ilic-Helms J, Rohrer B, Wheeler J and Larson K 2013 Sparse imaging for fast electron microscopy *Computational Imaging XI* vol 8657 (Int. Society for Optics and Photonics) p 86570C
[249] Stevens A, Yang H, Carin L, Arslan I and Browning N D 2013 The potential for bayesian compressive sensing to significantly reduce electron dose in high-resolution STEM images *Microscopy* **63** 41–51
[250] Candes E and Romberg J 2007 Sparsity and incoherence in compressive sampling *Inverse Probl.* **23** 969
[251] Kovarik L, Stevens A, Liyu A and Browning N D 2016 Implementing an accurate and rapid sparse sampling approach for low-dose atomic resolution STEM imaging *Appl. Phys. Lett.* **109** 164102
[252] Béché A, Goris B, Freitag B and Verbeeck J 2016 Development of a fast electromagnetic beam blanker for compressed sensing in scanning transmission electron microscopy *Appl. Phys. Lett.* **108** 093103
[253] Li X, Dyck O, Kalinin S V and Jesse S 2018 Compressed sensing of scanning transmission electron microscopy (STEM) with nonrectangular scans *Microsc. Microanal.* **24** 623–33
[254] Sang X *et al* 2017 Precision controlled atomic resolution scanning transmission electron microscopy using spiral scan pathways *Sci. Rep.* **7** 43585
[255] Gandhare S and Karthikeyan B 2019 Survey on FPGA architecture and recent applications *2019 Int. Conf. on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)* (IEEE) pp 1–4
[256] Qiao M, Meng Z, Ma J and Yuan X 2020 Deep learning for video compressive sensing *APL Photonics* **5** 030801
[257] Wu Y, Rosca M and Lillicrap T 2019 Deep compressed sensing (arXiv:1905.06723)
[258] Adler A, Boublil D and Zibulevsky M 2017 Block-based compressed sensing of images via deep learning *2017 IEEE 19th Int. Workshop on Multimedia Signal Processing (MMSP)* (IEEE) pp 1–6
[259] de Haan K, Ballard Z S, Rivenson Y, Wu Y and Ozcan A 2019 Resolution enhancement in scanning electron microscopy using deep learning *Sci. Rep.* **9** 1–7
[260] Gao Z, Ma W, Huang S, Hua P and Lan C 2020 Deep learning for super-resolution in a field emission scanning electron microscope *Artif. Intell.* **1** 1–10
[261] Ede J M and Beanland R 2020 Adaptive learning rate clipping stabilizes learning *Mach. Learn.: Sci. Technol.* **1** 015011
[262] Suveer A, Gupta A, Kylberg G and Sintorn I-M 2019 Super-resolution reconstruction of transmission electron microscopy images using deep learning *2019 IEEE 16th Int. Symp. on Biomedical Imaging* (IEEE) pp 548–51
[263] Ahmed M W and Abdulla A A 2020 Quality improvement for exemplar-based image inpainting using a modified searching mechanism *UHD J. Sci. Technol.* **4** 1–8
[264] Pinjarkar A V and Tuptewar D 2019 Robust exemplar-based image and video inpainting for object removal and region filling *Computing, Communication and Signal Processing* (Berlin: Springer) pp 817–25
[265] Zhang N, Ji H, Liu L and Wang G 2019 Exemplar-based image inpainting using angle-aware patch matching *EURASIP J. Image Video Process.* **2019** 70
[266] Criminisi A, Pérez P and Toyama K 2004 Region filling and object removal by exemplar-based image inpainting *IEEE Trans. Image Process.* **13** 1200–12
[267] Lu M and Niu S 2020 A detection approach using LSTM-CNN for object removal caused by exemplar-based image inpainting *Electronics* **9** 858
[268] Telea A 2004 An image inpainting technique based on the fast marching method *J. Graph. Tools* **9** 23–34
[269] Bertalmio M, Bertozzi A L and Sapiro G 2001 Navier–Stokes, fluid dynamics and image and video inpainting *Proc. 2001 IEEE Computer Conf. on Computer Vision and Pattern Recognition* vol 1 (IEEE) p I
[270] He T *et al* 2019 Bag of tricks for image classification with convolutional neural networks *Proc. Conf. on Computer Vision and Pattern Recognition* pp 558–67
[271] Sun Y, Xue B, Zhang M and Yen G G 2019 Evolving deep convolutional neural networks for image classification *IEEE Trans. Evol. Comput.* **24** 394–407
[272] Rawat W and Wang Z 2017 Deep convolutional neural networks for image classification: a comprehensive review *Neural Comput.* **29** 2352–449
[273] Druzhkov P N and Kustikova V D 2016 A survey of deep learning methods and software tools for image classification and object detection *Pattern Recognit. Image Anal.* **26** 9–15
[274] Yokoyama Y *et al* 2020 Development of a deep learning-based method to identify 'good' regions of a cryo-electron microscopy grid *Biophys. Rev.* **12** 349–54
[275] Sanchez-Garcia R, Segura J, Maluenda D, Sorzano C and Carazo J 2020 Micrographcleaner: a python package for cryo-em micrograph cleaning using deep learning *J. Struct. Biol.* 107498
[276] Aguiar J, Gong M, Unocic R, Tasdizen T and Miller B 2019 Decoding crystallography from high-resolution electron imaging and diffraction datasets with deep learning *Sci. Adv.* **5** eaaw1949
[277] Vasudevan R K *et al* 2018 Mapping mesoscopic phase evolution during e-beam induced transformations via deep learning of atomically resolved images *npj Comput. Mater.* **4** 30
[278] Avramov T K *et al* 2019 Deep learning for validating and estimating resolution of cryo-electron microscopy density maps *Molecules* **24** 1181
[279] Koch G, Zemel R and Salakhutdinov R 2015 Siamese neural networks for one-shot image recognition *ICML Deep Learning Workshop* vol 2 (Lille)
[280] Chopra S, Hadsell R and LeCun Y 2005 Learning a similarity metric discriminatively, with application to face verification *2005 IEEE Computer Conf. on Computer Vision and Pattern Recognition (CVPR'05)* vol 1 (IEEE) pp 539–46
[281] Bromley J, Guyon I, LeCun Y, Säckinger E and Shah R 1994 Signature verification using a 'siamese' time delay neural network *Advances in Neural Information Processing Systems* pp 737–44

[282] Cai Q, Pan Y, Yao T, Yan C and Mei T 2018 Memory matching networks for one-shot image recognition *Proc. Conf. on Computer Vision and Pattern Recognition* pp 4080–8
[283] Li X *et al* 2019 Predicting the effective mechanical property of heterogeneous materials by image based modeling and deep learning *Comput. Methods Appl. Mech. Eng.* **347** 735–53
[284] Sanchez-Garcia R, Segura J, Maluenda D, Carazo J M and Sorzano C O S 2018 Deep consensus, a deep learning-based approach for particle pruning in cryo-electron microscopy *IUCrJ* **5** 854–65
[285] Wang F *et al* 2016 DeepPicker: a deep learning approach for fully automated particle picking in cryo-EM *J. Struct. Biol.* **195** 325–36
[286] George B *et al* 2020 CASSPER: a semantic segmentation based particle picking algorithm for single particle cryo-electron microscopy *bioRxiv*
[287] Roberts G *et al* 2019 Deep learning for semantic segmentation of defects in advanced stem images of steels *Sci. Rep.* **9** 1–12
[288] Madsen J *et al* 2018 A deep learning approach to identify local structures in atomic-resolution transmission electron microscopy images *Adv. Theory Simul.* **1** 1800037
[289] Ziatdinov M *et al* 2017 Deep learning of atomically resolved scanning transmission electron microscopy images: chemical identification and tracking local transformations *ACS Nano* **11** 12742–52
[290] Ziatdinov M *et al* 2019 Building and exploring libraries of atomic defects in graphene: scanning transmission electron and scanning tunneling microscopy study *Sci. Adv.* **5** eaaw8989
[291] Meyer J C *et al* 2008 Direct imaging of lattice atoms and topological defects in graphene membranes *Nano Lett.* **8** 3582–6
[292] Meyer J C *et al* 2011 Experimental analysis of charge redistribution due to chemical bonding by high-resolution transmission electron microscopy *Nat. Mater.* **10** 209–15
[293] He X *et al* 2014 *In situ* atom scale visualization of domain wall dynamics in VO_2 insulator-metal phase transition *Sci. Rep.* **4** 6544
[294] Nagao K, Inuzuka T, Nishimoto K and Edagawa K 2015 Experimental observation of quasicrystal growth *Phys. Rev. Lett.* **115** 075501
[295] Li X *et al* 2017 Direct observation of the layer-by-layer growth of ZnO nanopillar by *in situ* high resolution transmission electron microscopy *Sci. Rep.* **7** 40911
[296] Schneider S, Surrey A, Pohl D, Schultz L and Rellinghaus B 2014 Atomic surface diffusion on Pt nanoparticles quantified by high-resolution transmission electron microscopy *Micron* **63** 52–6
[297] Hussaini Z, Lin P A, Natarajan B, Zhu W and Sharma R 2018 Determination of atomic positions from time resolved high resolution transmission electron microscopy images *Ultramicroscopy* **186** 139–45
[298] Pham D L, Xu C and Prince J L 2000 Current methods in medical image segmentation *Annu. Rev. Biomed. Eng.* **2** 315–37
[299] Mesejo P, Valsecchi A, Marrakchi-Kacem L, Cagnoni S and Damas S 2015 Biomedical image segmentation using geometric deformable models and metaheuristics *Comput. Med. Imaging Graph.* **43** 167–78
[300] Zheng Y, Jeon B, Xu D, Wu Q M and Zhang H 2015 Image segmentation by generalized hierarchical fuzzy c-means algorithm *J. Intell. Fuzzy Syst.* **28** 961–73
[301] Hao S, Zhou Y and Guo Y 2020 A brief survey on semantic segmentation with deep learning *Neurocomputing* **406** 302–21
[302] Sultana F, Sufian A and Dutta P 2020 Evolution of image segmentation using deep convolutional neural network: a survey *Knowl.-Based Syst.* **201–202** 106062
[303] Minaee S *et al* 2020 Image segmentation using deep learning: a survey (arXiv:2001.05566)
[304] Guo Y, Liu Y, Georgiou T and Lew M S 2018 A review of semantic segmentation using deep neural networks *Int. J. Multimedia Inf. Retr.* **7** 87–93
[305] Chen L-C, Zhu Y, Papandreou G, Schroff F and Adam H 2018 Encoder–decoder with atrous separable convolution for semantic image segmentation *Proc. Conf. on Computer Vision (ECCV)* pp 801–18
[306] Chen L-C, Papandreou G, Schroff F and Adam H 2017 Rethinking atrous convolution for semantic image segmentation (arXiv:1706.05587)
[307] Badrinarayanan V, Kendall A and Cipolla R 2017 SegNet: a deep convolutional encoder–decoder architecture for image segmentation *IEEE Trans. Pattern Anal. Mach. Intell.* **39** 2481–95
[308] Ronneberger O, Fischer P and Brox T 2015 U-Net: convolutional networks for biomedical image segmentation *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention* (Springer) pp 234–41
[309] Yi J, Yuan Z and Peng J 2020 Adversarial-prediction guided multi-task adaptation for semantic segmentation of electron microscopy images *2020 IEEE 17th Int. Symp. on Biomedical Imaging (ISBI)* (IEEE) pp 1205–8
[310] Khadangi A, Boudier T and Rajagopal V 2020 EM-net: deep learning for electron microscopy image segmentation *bioRxiv*
[311] Roels J and Saeys Y 2019 Cost-efficient segmentation of electron microscopy images using active learning (arXiv:1911.05548)
[312] Yu Z X *et al* 2020 High-Throughput, algorithmic determination of pore parameters from electron microscopy *Comput. Mater. Sci.* **171** 109216
[313] Fakhry A, Zeng T and Ji S 2016 Residual deconvolutional networks for brain electron microscopy image segmentation *IEEE Trans. Med. Imaging* **36** 447–56
[314] Urakubo H, Bullmann T, Kubota Y, Oba S and Ishii S 2019 UNI-EM: an environment for deep neural network-based automated segmentation of neuronal electron microscopic images *Sci. Rep.* **9** 1–9
[315] Roberts G *et al* 2019 DefectNet—a deep convolutional neural network for semantic segmentation of crystallographic defects in advanced microscopy images *Microsc. Microanal.* **25** 164–5
[316] Ibtehaz N and Rahman M S 2020 MultiResUNet: rethinking the U-Net architecture for multimodal biomedical image segmentation *Neural Netw.* **121** 74–87
[317] Groschner C K, Choi C and Scott M 2020 Methodologies for successful segmentation of HRTEM images via neural network (arXiv:2001.05022)
[318] Horwath J P, Zakharov D N, Megret R and Stach E A 2019 Understanding important features of deep learning models for transmission electron microscopy image segmentation (arXiv:1912.06077)
[319] Chen M *et al* 2017 Convolutional neural networks for automated annotation of cellular cryo-electron tomograms *Nat. Methods* **14** 983
[320] Feng D *et al* 2020 Deep multi-modal object detection and semantic segmentation for autonomous driving: datasets, methods and challenges *IEEE Trans. Intell. Transp. Syst.* (https://doi.org/10.1109/TITS.2020.2972974)
[321] Yang K, Bi S and Dong M 2020 Lightningnet: fast and accurate semantic segmentation for autonomous driving based on 3D LIDAR point cloud *2020 IEEE Int. Conf. on Multimedia and Expo* (IEEE) pp 1–6

[322] Hofmarcher M *et al* 2019 Visual scene understanding for autonomous driving using semantic segmentation *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Springer) pp 285–96
[323] Blum H, Sarlin P-E, Nieto J, Siegwart R and Cadena C 2019 Fishyscapes: a benchmark for safe semantic segmentation in autonomous driving *Proc. IEEE Int. Conf. on Computer Vision Workshops*
[324] Zhou W, Berrio J S, Worrall S and Nebot E 2019 Automated evaluation of semantic segmentation robustness for autonomous driving *IEEE Trans. Intell. Transp. Syst.* **21** 1951–63
[325] Pfisterer K J *et al* 2019 Fully-automatic semantic segmentation for food intake tracking in long-term care homes (arXiv:1910.11250)
[326] Aslan S, Ciocca G and Schettini R 2018 Semantic food segmentation for automatic dietary monitoring *2018 IEEE 8th Int. Conf. on Consumer Electronics-Berlin* (IEEE) pp 1–6
[327] Ghosh S, Ray N, Boulanger P, Punithakumar K and Noga M 2020 Automated left atrial segmentation from magnetic resonance image sequences using deep convolutional neural network with autoencoder *2020 IEEE 17th Int. Symp. on Biomedical Imaging* (IEEE) pp 1756–60
[328] Memis A, Varli S and Bilgili F 2020 Semantic segmentation of the multiform proximal femur and femoral head bones with the deep convolutional neural networks in low quality mri sections acquired in different mri protocols *Comput. Med. Imaging Graph.* **81** 101715
[329] Duran A, Jodoin P-M and Lartizien C 2020 Prostate cancer semantic segmentation by gleason score group in mp-MRI with self attention model on the peripheral zone *Medical Imaging with Deep Learning*
[330] Bevilacqua V *et al* 2019 A comparison between two semantic deep learning frameworks for the autosomal dominant polycystic kidney disease segmentation based on magnetic resonance images *BMC Med. Inform. Decis. Mak.* **19** 1–12
[331] Liu F *et al* 2018 Deep convolutional neural network and 3D deformable approach for tissue segmentation in musculoskeletal magnetic resonance imaging *Magn. Reson. Med.* **79** 2379–91
[332] Taghanaki S A, Abhishek K, Cohen J P, Cohen-Adad J and Hamarneh G 2020 Deep semantic segmentation of natural and medical images: a review *Artif. Intell. Rev.*
[333] Tajbakhsh N *et al* 2020 Embracing imperfect datasets: a review of deep learning solutions for medical image segmentation *Med. Image Anal.* **63** 101693
[334] Du G, Cao X, Liang J, Chen X and Zhan Y 2020 Medical image segmentation based on U-Net: a review *J. Imaging Sci. Technol.* **64** 20508–1
[335] Yang X *et al* 2020 Hybrid attention for automatic segmentation of whole fetal head in prenatal ultrasound volumes *Comput. Methods Programs Biomed.* **194** 105519
[336] Wang X *et al* 2019 Joint segmentation and landmark localization of fetal femur in ultrasound volumes *2019 IEEE EMBS Int. Conf. on Biomedical and Health Informatics (BHI)* (IEEE) pp 1–5
[337] Venturini L, Papageorghiou A T, Noble J A and Namburete A I 2019 Multi-task CNN for structural semantic segmentation in 3D fetal brain ultrasound *Conf. on Medical Image Understanding and Analysis* (Springer) pp 164–73
[338] Yang X *et al* 2018 Towards automated semantic segmentation in prenatal volumetric ultrasound *IEEE Trans. Med. Imaging* **38** 180–93
[339] Tasar O, Tarabalka Y, Giros A, Alliez P and Clerc S 2020 StandardGAN: multi-source domain adaptation for semantic segmentation of very high resolution satellite images by data standardization *Proc. IEEE/Conf. on Computer Vision and Pattern Recognition Workshops* pp 192–3
[340] Barthakur M and Sarma K K 2020 Deep learning based semantic segmentation applied to satellite image *Data Visualization and Knowledge Engineering* (Springer) pp 79–107
[341] Wu M, Zhang C, Liu J, Zhou L and Li X 2019 Towards accurate high resolution satellite image semantic segmentation *IEEE Access* **7** 55609–19
[342] Wurm M, Stark T, Zhu X X, Weigand M and Taubenböck H 2019 Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks *ISPRS J. Photogramm. Remote Sens.* **150** 59–69
[343] Zhou L, Zhang C and Wu M 2018 D-LinkNet: linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction *CVPR Workshops* pp 182–6
[344] Joyce T, Chartsias A and Tsaftaris S A 2018 Deep multi-class segmentation without ground-truth labels *1st Conf. on Medical Imaging With Deep Learning*
[345] Araslanov N and Roth S 2020 Single-stage semantic segmentation from image labels *Proc. IEEE/Conf. on Computer Vision and Pattern Recognition* pp 4253–62
[346] Chen Z, Tian Z, Li X, Zhang Y and Dormer J D 2020 Exploiting confident information for weakly supervised prostate segmentation based on image-level labels *Medical Imaging 2020: Image-Guided Procedures, Robotic Interventions and Modeling* vol 11315 (Int. Society for Optics and Photonics) p 1131523
[347] Jing L, Chen Y and Tian Y 2019 Coarse-to-fine semantic segmentation from image-level labels *IEEE Trans. Image Process.* **29** 225–36
[348] Oh S J *et al* 2017 Exploiting saliency for object segmentation from image level labels *Conf. on Computer Vision and Pattern Recognition* (IEEE) pp 5038–47
[349] Ede J M, Peters J J P, Sloan J and Beanland R 2020 Exit wavefunction reconstruction from single transmission electron micrographs with deep learning (arXiv:2001.10938)
[350] Frabboni S, Gazzadi G C and Pozzi G 2007 Young's double-slit interference experiment with electrons *Am. J. Phys.* **75** 1053–5
[351] Matteucci G and Beeli C 1998 An experiment on electron wave-particle duality including a planck constant measurement *Am. J. Phys.* **66** 1055–9
[352] Lehmann M and Lichte H 2002 Tutorial on off-axis electron holography *Microsc. Microanal.* **8** 447–66
[353] Tonomura A 1987 Applications of electron holography *Rev. Mod. Phys.* **59** 639
[354] Lentzen M and Urban K 2000 Reconstruction of the projected crystal potential in transmission electron microscopy by means of a maximum-likelihood refinement algorithm *Acta Crystallogr.* A **56** 235–47
[355] Auslender A, Halabi M, Levi G, Diéguez O and Kohn A 2019 Measuring the mean inner potential of Al_2O_3 sapphire using off-axis electron holography *Ultramicroscopy* **198** 18–25
[356] Fu Q, Lichte H and Völkl E 1991 Correction of aberrations of an electron microscope by means of electron holography *Phys. Rev. Lett.* **67** 2319
[357] McCartney M R and Gajdardziska-Josifovska M 1994 Absolute measurement of normalized thickness, t/λ_i, from off-axis electron holography *Ultramicroscopy* **53** 283–9

[358] Park H S *et al* 2014 Observation of the magnetic flux and three-dimensional structure of skyrmion lattices by electron holography *Nat. Nanotechnol.* **9** 337–42
[359] Dunin-Borkowski R E *et al* 2004 Off-axis electron holography of magnetic nanowires and chains, rings and planar arrays of magnetic nanoparticles *Microsc. Res. Tech.* **64** 390–402
[360] Lubk A *et al* 2016 Fundamentals of focal series inline electron holography *Advances in Imaging and Electron Physics* vol 197 (Elsevier) pp 105–47
[361] Koch C T 2014 Towards full-resolution inline electron holography *Micron* **63** 69–75
[362] Haigh S J, Jiang B, Alloyeau D, Kisielowski C and Kirkland A I 2013 Recording low and high spatial frequencies in exit wave reconstructions *Ultramicroscopy* **133** 26–34
[363] Koch C T and Lubk A 2010 Off-Axis and inline electron holography: a quantitative comparison *Ultramicroscopy* **110** 460–71
[364] Van Dyck D, de Beeck M O and Coene W 1994 Object wavefunction reconstruction in high resolution electron microscopy *Proc. 1st Int. Conf. on Image Processing* vol 3 (IEEE) pp 295–8
[365] Ozsoy-Keskinbora C, Boothroyd C, Dunin-Borkowski R, Van Aken P and Koch C 2014 Hybridization approach to in-line and off-axis (electron) holography for superior resolution and phase sensitivity *Sci. Rep.* **4** 1–10
[366] Rivenson Y, Zhang Y, Günaydin H, Teng D and Ozcan A 2018 Phase recovery and holographic image reconstruction using deep learning in neural networks *Light: Sci. Appl.* **7** 17141
[367] Wu Y *et al* 2018 Extended depth-of-field in holographic imaging using deep-learning-based autofocusing and phase recovery *Optica* **5** 704–10
[368] Sinha A, Lee J, Li S and Barbastathis G 2017 Lensless computational imaging through deep learning *Optica* **4** 1117–25
[369] Beach M J *et al* 2018 QuCumber: wavefunction reconstruction with neural networks (arXiv:1812.09329)
[370] Dral P O 2020 Quantum chemistry in the age of machine learning *J. Phys. Chem. Lett.* **11** 2336–47
[371] Liu X *et al* 2020 Deep learning for Feynman's path integral in strong-field time-dependent dynamics *Phys. Rev. Lett.* **124** 113202
[372] Bharti K, Haug T, Vedral V and Kwek L-C 2020 Machine learning meets quantum foundations: a brief survey (arXiv:2003.11224)
[373] Carleo G *et al* 2019 NetKet: a machine learning toolkit for many-body quantum systems (arXiv:1904.00031)
[374] Schütt K, Gastegger M, Tkatchenko A, Müller K-R and Maurer R J 2019 Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions *Nat. Commun.* **10** 1–10
[375] Laanait N, He Q and Borisevich A Y 2019 Reconstruction of 3-D atomic distortions from electron microscopy with deep learning (arXiv:1902.06876)
[376] Morgan A J, Martin A V, D'Alfonso A J, Putkunz C T and Allen L J 2011 Direct exit-wave reconstruction from a single defocused image *Ultramicroscopy* **111** 1455–60
[377] Martin A and Allen L 2008 Direct retrieval of a complex wave from its diffraction pattern *Opt. Commun.* **281** 5114–21
[378] Schlitz M 2018 Science without publication paywalls a preamble to: coalition s for the realisation of full and immediate open access *Science Europe*
[379] Coalition of European Funders Announces 'Plan S' to require full OA, cap APCs, and disallow publication in hybrid journals 2018 SPARC (available at: https://sparcopen.org/news/2018/coalition-european-funders-announces-plan-s)
[380] cOAlition S 2020 Plan S: making full and immediate open access a reality (available at: www.coalition-s.org)
[381] Banks G C *et al* 2019 Answers to 18 questions about open science practices *J. Bus. Psychol.* **34** 257–70
[382] Shi R *et al* 2020 FTDL: an FPGA-tailored architecture for deep learning systems *FPGA* p 320
[383] Kaarmukilan S *et al* 2020 FPGA based deep learning models for object detection and recognition comparison of object detection comparison of object detection models using FPGA *2020 Fourth Int. Conf. on Computing Methodologies and Communication (ICCMC)* (IEEE) pp 471–4
[384] Wang T, Wang C, Zhou X and Chen H 2019 An overview of FPGA based deep learning accelerators: challenges and opportunities *2019 IEEE 21st Int. Conf. on High Performance Computing and Communications; IEEE 17th Int. Conf. on Smart City; IEEE 5th Int. Conf. on Data Science and Systems (HPCC/SmartCity/DSS)* (IEEE) pp 1674–81
[385] Guo K, Zeng S, Yu J, Wang Y and Yang H 2019 [DL] a survey of fpga-based neural network inference accelerators *ACM Trans. Reconfigurable Technol. Syst.* **12** 1–26
[386] Cano A 2018 A survey on graphic processing unit computing for large-scale data mining *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **8** e1232
[387] Nvidia 2017 Tesla V100 GPU architecture whitepaper (available at: https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf)
[388] Gaster B R 2013 *Heterogeneous Computing With Opencl* 2nd edn (Elsevier/Morgan Kaufmann)
[389] Gordienko Y *et al* 2020 Scaling analysis of specialized tensor processing architectures for deep learning models *Deep Learning: Concepts and Architectures* (Springer) pp 65–99
[390] Jouppi N, Young C, Patil N and Patterson D 2018 Motivation for and evaluation of the first tensor processing unit *IEEE Micro* **38** 10–19
[391] Jouppi N P *et al* 2017 In-datacenter performance analysis of a tensor processing unit *Proc. 44th Annual International Symposium on Computer Architecture* pp 1–12
[392] Mattson P *et al* 2020 MLPerf training benchmark (arXiv:1910.01500)
[393] MLPerf: Fair and Useful benchmarks for measuring training and inference performance of ML hardware, software, and services 2020 (available at: https://mlperf.org)
[394] Wang Y E, Wei G-Y and Brooks D 2019 Benchmarking TPU, GPU, and CPU platforms for deep learning (arXiv:1907.10701)
[395] Wang Y *et al* 2019 Performance and power evaluation of AI accelerators for training deep learning models (arXiv:1909.06842)
[396] Li F, Ye Y, Tian Z and Zhang X 2019 Cpu versus gpu: which can perform matrix computation faster—performance comparison for basic linear algebra subprograms *Neural Comput. Appl.* **31** 4353–65
[397] Awan A A, Subramoni H and Panda D K 2017 An in-depth performance characterization of CPU-and GPU-based DNN training on modern architectures *Proc. Machine Learning on HPC Environments* pp 1–8
[398] Nurvitadhi E *et al* 2017 Can FPGAs beat GPUs in accelerating next-generation deep neural networks? *Proc. 2017 ACM/SIGDA Int. Symp. on Field-Programmable Gate Arrays* pp 5–14
[399] GPU vs FPGA Performance Comparison 2016 Berten digital signal processing (available at: www.bertendsp.com/pdf/whitepaper/BWP001_GPU_vs_FPGA_Performance_Comparison_v1.0.pdf)
[400] Nangia R and Shukla N K 2018 Resource utilization optimization with design alternatives in FPGA based arithmetic logic unit architectures *Proc. Comput. Sci.* **132** 843–8

[401] Grover N and Soni M 2014 Design of FPGA based 32-bit floating point arithmetic unit and verification of its VHDL code using MATLAB *Int. J. Inf. Eng. Electron. Bus.* **6** 1
[402] Dolbeau R 2018 Theoretical peak FLOPS per instruction set: a tutorial *J. Supercomput.* **74** 1341–77
[403] Strubell E, Ganesh A and McCallum A 2019 Energy and policy considerations for deep learning in NLP (arXiv:1906.02243)
[404] Nelson M J and Hoover A K 2020 Notes on using google colaboratory in AI education *Proc. 2020 Conf. on Innovation and Technology in Computer Science Education* pp 533–4
[405] Bisong E 2019 Google colaboratory *Building Machine Learning and Deep Learning Models on Google Cloud Platform* (Springer) pp 59–64
[406] Tutorialspoint 2019 Colab tutorial (available at: www.tutorialspoint.com/google_colab/google_colab_tutorial.pdf)
[407] Carneiro T *et al* 2018 Performance analysis of google colaboratory as a tool for accelerating deep learning applications *IEEE Access* **6** 61677–85
[408] Kaggle Documentation 2020 (available at: www.kaggle.com/docs)
[409] Kalinin S V, Vasudevan R K and Ziatdinov M 2020 Decoding the relationship between domain structure and functionality in ferroelectrics via hidden latent variables (arXiv:2006.01374)
[410] Green O 2019 How to install a new graphics card—from hardware to drivers Help Desk Geek (available at: https://helpdeskgeek.com/how-to/how-to-install-a-new-graphics-card-from-hardware-to-drivers)
[411] Ryan T 2017 How to install a graphics card PC World (available at: www.pcworld.com/article/2913370/how-to-install-a-graphics-card.html)
[412] Radecic D 2020 An utterly simple guide on installing Tensorflow-GPU 2.0 on Windows 10 Towards Data Science (available at: https://towardsdatascience.com/an-utterly-simple-guide-on-installing-tensorflow-gpu-2-0-on-windows-10-198368dc07a1)
[413] Varile M 2019 Train neural networks using AMD GPU and Keras Towards Data Science (available at: https://towardsdatascience.com/train-neural-networks-Using-amd-gpus-and-keras-37189c453878)
[414] Tim Dettmers 2018 A full hardware guide to deep learning (available at: https://timdettmers.com/2018/12/16/deep-learning-hardware-guide)
[415] Chetlur S *et al* 2014 cuDNN: efficient primitives for deep learning (arXiv:1410.0759)
[416] List of Cloud Services for Deep Learning 2020 (available at: https://github.com/zszazi/Deep-learning-in-cloud)
[417] Marozzo F 2019 Infrastructures for high-performance computing: cloud infrastructures *Encyclopedia of Bioinformatics and Computational Biology* pp 240–6
[418] Joshi N and Shah S 2019 A comprehensive survey of services provided by prevalent cloud computing environments *Smart Intelligent Computing and Applications* (Berlin: Springer) pp 413–24
[419] Gupta A, Goswami P, Chaudhary N and Bansal R 2020 Deploying an application using google cloud platform *2020 2nd Int. Conf. on Innovative Mechanisms for Industry Applications (ICIMIA)* (IEEE) pp 236–9
[420] Ooi B C *et al* 2015 SINGA: a distributed deep learning platform *Proc. 23rd ACM Conf. on Multimedia* pp 685–8
[421] Apache SINGA License 2020 (available at: https://github.com/apache/singa/blob/master/LICENSE)
[422] Dai J J *et al* 2019 BigDL: a distributed deep learning framework for big data *Proc. Symp. on Cloud Computing* pp 50–60
[423] BigDL License 2020 (available at: https://github.com/intel-analytics/BigDL/blob/master/LICENSE)
[424] Jia Y 2014 *et al* Caffe: convolutional architecture for fast feature embedding *Proc. 22nd ACM Int. Conf. on Multimedia* pp 675–8
[425] Synced 2004 Caffe2 Merges with PyTorch. Medium (available at: https://medium.com/@Synced/caffe2-merges-with-pytorch-a89c70ad9eb7)
[426] Caffe License 2017 (available at: https://github.com/BVLC/caffe/blob/master/LICENSE)
[427] Tokui S, Oono K, Hido S and Clayton J 2015 Chainer: a next-generation open source framework for deep learning *Proc. of Workshop on Machine Learning Systems (LearningSys) in the 29th Conf. on Neural Information Processing Systems (NIPS)* vol 5 pp 1–6
[428] Chainer License 2020 (available at: https://docs.chainer.org/en/stable/license.html)
[429] Gibson A *et al* 2016 Deeplearning4j: distributed, open-source deep learning for Java and Scala on Hadoop and Spark Towards Data Science (available at: https://deeplearning4j.org)
[430] Deeplearning4j License 2020 (available at: https://github.com/eclipse/deeplearning4j/blob/master/LICENSE)
[431] King D E 2009 Dlib-ml: a machine learning toolkit *J. Mach. Learn. Res.* **10** 1755–8
[432] Dlib C++ Library 2020 (available at: http://dlib.net)
[433] Dlib License 2020 (available at: https://github.com/davisking/dlib/blob/master/dlib/LICENSE.txt)
[434] Innes M 2018 Flux: elegant machine learning with Julia *J. Open Source Softw.* **3** 602
[435] Flux License 2020 (available at: https://github.com/FluxML/Flux.jl/blob/master/LICENSE.md)
[436] Beale M, Hagan M and Demuth H 2020 PDF documentation: matlab deep learning toolbox user's guide (available at: https://uk.mathworks.com/help/deeplearning)
[437] MATLAB License 2020 (available at: https://mathworks.com/pricing-licensing.html)
[438] Seide F 2017 Keynote: the computer science behind the microsoft cognitive toolkit: an open source large-scale deep learning toolkit for windows and linux *2017 IEEE/ACM Int. Symp. on Code Generation and Optimization (CGO)* (IEEE) p xi
[439] CNTK License 2020 (available at: https://github.com/microsoft/CNTK/blob/master/LICENSE.md)
[440] Chen T *et al* 2015 MXNet: a flexible and efficient machine learning library for heterogeneous distributed systems (arXiv:1512.01274)
[441] MXNet License 2020 (available at: https://github.com/apache/incubator-mxnet/blob/master/LICENSE)
[442] OpenNN 2020 (available at: www.opennn.net)
[443] OpenNN License 2020 (available at: https://github.com/Artelnics/OpenNN/blob/master/LICENSE.txt)
[444] Ma Y, Yu D, Wu T and Wang H 2019 PaddlePaddle: an open-source deep learning platform from industrial practice *Front. Data Comput.* **1** 105–15
[445] PaddlePaddle License 2020 (available at: https://github.com/PaddlePaddle/Paddle/blob/develop/LICENSE)
[446] Paszke A *et al* 2019 PyTorch: an imperative style, high-performance deep learning library *Advances in Neural Information Processing Systems* pp 8024–35
[447] PyTorch License 2020 (available at: https://github.com/pytorch/pytorch/blob/master/LICENSE)
[448] Abadi M *et al* 2016 TensorFlow: a system for large-scale machine learning *12th Symp. on Operating Systems Design and Implementation (OSDI 16)* pp 265–83
[449] Abadi M *et al* 2016 TensorFlow: large-scale machine learning on heterogeneous distributed systems (arXiv:1603.04467)
[450] TensorFlow License 2020 (available at: https://github.com/tensorflow/tensorflow/blob/master/LICENSE)

[451] Team T T D *et al* 2016 Theano: a python framework for fast computation of mathematical expressions (arXiv:1605.02688)
[452] Ketkar N 2017 Introduction to theano *Deep Learning With Python* (Berlin: Springer) pp 35–61
[453] Theano License 2020 (available at: https://github.com/Theano/Theano/blob/master/doc/LICENSE.txt)
[454] Collobert R, Bengio S and Mariéthoz J 2002 Torch: a modular machine learning software library *Technical Report* Idiap
[455] Torch License 2020 (available at: https://github.com/torch/torch7/blob/master/COPYRIGHT.txt)
[456] Mathematica Neural Networks Documentation 2020 (available at: https://reference.wolfram.com/language/guide/NeuralNetworks.html)
[457] Mathematica Licenses 2020 (available at: www.wolfram.com/legal)
[458] Li M *et al* 2020 The deep learning compiler: a comprehensive survey (arXiv:2002.03794)
[459] Nguyen G *et al* 2019 Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey *Artif. Intell. Rev.* **52** 77–124
[460] Dai W and Berleant D 2019 Benchmarking contemporary deep learning hardware and frameworks: a survey of qualitative metrics *2019 IEEE First Int. Conf. on Cognitive Machine Intelligence (CogMI)* (IEEE) pp 148–55
[461] Kharkovyna O 2019 Top 10 best deep learning frameworks in 2019 Towards Data Science (available at: https://towardsdatascience.com/top-10-best-deep-learning-frameworks-in-2019-5ccb90ea6de)
[462] Zacharias J, Barz M and Sonntag D 2018 A survey on deep learning toolkits and libraries for intelligent user interfaces (arXiv:1803.04818)
[463] Parvat A, Chavan J, Kadam S, Dev S and Pathak V 2017 A survey of deep-learning frameworks *2017 Int. Conf. on Inventive Systems and Control (ICISC)* (IEEE) pp 1–7
[464] Erickson B J, Korfiatis P, Akkus Z, Kline T and Philbrick K 2017 Toolkits and libraries for deep learning *J. Digit. Imaging* **30** 400–5
[465] Baydin A G, Pearlmutter B A, Radul A A and Siskind J M 2017 *J. Mach. Learn. Res.* **18** 5595–637
[466] Barham P and Isard M 2019 Machine learning systems are stuck in a rut *Proc. Workshop on Hot Topics in Operating Systems* pp 177–83
[467] Afif M, Said Y and Atri M 2020 Computer vision algorithms acceleration using graphic processors NVIDIA CUDA *Cluster Computing* pp 1–13
[468] Cook S 2012 *Cuda Programming: A Developer's Guide to Parallel Computing With Gpus* 1st edn (San Francisco, CA: Morgan Kaufmann Publishers Inc.)
[469] Nickolls J, Buck I, Garland M and Skadron K 2008 Scalable parallel programming with CUDA *Queue* **6** 40–53
[470] Jordà M, Valero-Lara P and Peña A J 2019 Performance evaluation of cudnn convolution algorithms on NVIDIA volta GPUs *IEEE Access* **7** 70461–73
[471] de Supinski B R *et al* 2018 The ongoing evolution of openMP *Proc. IEEE* **106** 2004–19
[472] Dagum L and Menon R 1998 OpenMP: an industry standard API for shared-memory programming *IEEE Comput. Sci. Eng.* **5** 46–55
[473] He H 2019 The state of machine learning frameworks in 2019 The Gradient (available at: https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominates-industry)
[474] Papers With Code: Trends 2020 (available at: https://paperswithcode.com/trends)
[475] TensorFlow Libraries and Extensions 2020 (available at: www.tensorflow.org/resources/libraries-extensions)
[476] Chollet F *et al* 2020 Keras (available at: https://keras.io)
[477] Sonnet repository 2020 (available at: https://github.com/deepmind/sonnet)
[478] Vaswani A *et al* 2018 Tensor2tensor for neural machine translation (arXiv:1803.07416)
[479] Tang Y 2016 TF.Learn: tensorflow's high-level module for distributed machine learning (arXiv:1612.04251)
[480] Damien A *et al* 2019 TFLearn repository (available at: https://github.com/tflearn/tflearn)
[481] TensorFlow Addons 2020 (available at: https://github.com/tensorflow/addons)
[482] Sergio G *et al* 2018 TF-Agents: a library for reinforcement learning in tensorflow (available at: https://github.com/tensorflow/agents)
[483] Castro P S, Moitra S, Gelada C, Kumar S and Bellemare M G 2018 Dopamine: a research framework for deep reinforcement learning (arXiv:1812.06110)
[484] McMahan B and Ramage D 2017 Federated learning: collaborative machine learning without centralized training data *Google Research Blog* **4**
[485] TensorFlow Federated 2018 (available at: https://github.com/tensorflow/federated)
[486] Caldas S *et al* 2018 LEAF: a benchmark for federated settings (arXiv:1812.01097)
[487] Dillon J V *et al* 2017 TensorFlow distributions (arXiv:1711.10604)
[488] Hessel M, Martic M, de Las Casas D and Barth-Maron G 2018 Open sourcing trfl: a library of reinforcement learning building blocks DeepMind Blog (available at: https://blog.paperspace.com/geometric-deep-learning-framework-comparison)
[489] Pedregosa F *et al* 2011 Scikit-learn: machine learning in python *J. Mach. Learn. Res.* **12** 2825–30
[490] ANNdotNET 2020 (available at: https://github.com/bhrnjica/anndotnet)
[491] Create ML Documentation 2020 (available at: https://developer.apple.com/documentation/createml)
[492] Deep Cognition 2020 (available at: https://deepcognition.ai)
[493] MathWorks Deep Network Designer 2020 (available at: https://uk.mathworks.com/help/deeplearning/ref/deepnetworkdesigner-app.html)
[494] DIGITS 2020 (available at: https://developer.nvidia.com/digits)
[495] ENNUI 2020 (available at: https://math.mit.edu/ennui)
[496] Expresso 2020 (available at: http://val.serc.iisc.ernet.in/expresso)
[497] Neural Designer: Data Science and Machine Learning Platform 2020 (available at: www.neuraldesigner.com)
[498] Witten I H, Frank E, Hall M A and Pal C J 2016 *Data Mining: Practical Machine Learning Tools and Techniques* (Morgan Kaufmann)
[499] Hall M *et al* 2009 The WEKA data mining software: an update *ACM SIGKDD Explorations Newsl.* **11** 10–18
[500] Holmes G, Donkin A and Witten I H 1994 WEKA: a machine learning workbench *Proc. of ANZIIS'94-Australian New Zealnd Intelligent Information Conf* (IEEE) pp 357–61
[501] Von Chamier L *et al* 2020 ZeroCostDL4Mic: an open platform to simplify access and use of deep-learning in microscopy *BioRxiv*
[502] Ye J C and Sung W K 2019 Understanding geometry of encoder–decoder CNNs (arXiv:1901.07647)

[503] Ye J C, Han Y and Cha E 2018 Deep convolutional framelets: a general deep learning framework for inverse problems *SIAM J. Imaging Sci.* **11** 991–1048
[504] Sutskever I, Vinyals O and Le Q V 2014 Sequence to sequence learning with neural networks *Advances in Neural Information Processing Systems* pp 3104–12
[505] List of Collections of Pretrained Models 2020 (available at: https://awesomeopensource.com/projects/pretrained-models)
[506] Model Zoo 2020 (available at: https://modelzoo.co)
[507] Open Neural Network Exchange 2020 (available at: https://onnx.ai)
[508] Bai J *et al* 2020 ONNX: open neural network exchange (available at: https://github.com/onnx/onnx)
[509] Shah S 2017 Microsoft and facebook's open AI ecosystem gains more support Engadget (available at: www.engadget.com/2017/10/11/microsoft-facebooks-ai-onxx-partners)
[510] Boyd E 2017 Microsoft and Facebook create open ecosystem for AI model interoperability Microsoft Azure Blog (available at: https://azure.microsoft.com/en-us/blog/microsoft-and-facebook-create-open-ecosystem-for-ai-model-interoperability)
[511] ONNX Model Zoo 2020 (available at: https://github.com/onnx/models)
[512] Gordon J 2018 Introducing tensorflow hub: a library for reusable machine learning modules in tensorflow medium (available at: https://tfhub.dev)
[513] TensorFlow Hub 2020 (available at: https://tfhub.dev)
[514] TensorFlow Model Garden 2020 (available at: https://github.com/tensorflow/models)
[515] Liang H, Fu W and Yi F 2019 A survey of recent advances in transfer learning *2019 IEEE 19th Int. Conf. on Communication Technology (ICCT)* (IEEE) pp 1516–23
[516] Zhuang F *et al* 2019 A comprehensive survey on transfer learning (arXiv:1911.02685)
[517] Tan C *et al* 2018 A survey on deep transfer learning *Int. Conf. on Artificial Neural Networks* (Springer) pp 270–9
[518] Marcelino P 2018 Transfer learning from pre-trained models Towards Data Science (available at: https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751)
[519] Weiss K, Khoshgoftaar T M and Wang D 2016 A survey of transfer learning *J. Big data* **3** 9
[520] Yosinski J, Clune J, Bengio Y and Lipson H 2014 How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems* pp 3320–8
[521] Da Silva F L, Warnell G, Costa A H R and Stone P 2020 Agents teaching agents: a survey on inter-agent transfer learning *Auton. Agent. Multi Agent Syst.* **34** 9
[522] Shermin T *et al* 2019 Enhanced transfer learning with imagenet trained classification layer *Pacific- Symp. on Image and Video Technology* (Springer) pp 142–55
[523] Ada S E, Ugur E and Akin H L 2019 Generalization in transfer learning (arXiv:1909.01331)
[524] The Khronos NNEF Working Group 2020 Neural network exchange format (available at: www.khronos.org/registry/NNEF)
[525] The HDF Group 2020 Hierarchical data format, Version 5 (available at: www.hdfgroup.org/HDF5)
[526] HDF5 for Python 2020 (available at: www.h5py.org)
[527] Somnath S, Smith C R, Laanait N, Vasudevan R K and Jesse S 2019 USID and pycroscopy—open source frameworks for storing and analyzing imaging and spectroscopy data *Microsc. Microanal.* **25** 220–1
[528] Pycroscopy Repository 2020 (available at: https://github.com/pycroscopy/pycroscopy)
[529] HyperSpy 2020 (available at: https://hyperspy.org)
[530] de la Peña F *et al* 2017 Electron microscopy (big and small) data analysis with the open source software package hyperspy *Microsc. Microanal.* **23** 214–15
[531] Rezk N M, Purnaprajna M, Nordström T and Ul-Abdin Z 2020 Recurrent neural networks: an embedded computing perspective *IEEE Access* **8** 57967–96
[532] Du K-L and Swamy M 2019 Recurrent neural networks *Neural Networks and Statistical Learning* (Springer) pp 351–71
[533] Yu Y, Si X, Hu C and Zhang J 2019 A review of recurrent neural networks: LSTM cells and network architectures *Neural Comput.* **31** 1235–70
[534] Choe Y J, Shin J and Spencer N 2017 Probabilistic interpretations of recurrent neural networks *Probabilistic Graphical Models*
[535] Choi M, Kim T and Kim J 2017 Awesome recurrent neural networks (available at: https://github.com/kjw0612/awesome-rnn)
[536] Lipton Z C, Berkowitz J and Elkan C 2015 A critical review of recurrent neural networks for sequence learning (arXiv:1506.00019)
[537] Hanin B and Rolnick D 2018 How to start training: the effect of initialization and architecture *Advances in Neural Information Processing Systems* pp 571–81
[538] Raschka S 2018 Model evaluation, model selection, and algorithm selection in machine learning (arXiv:1811.12808)
[539] Chollet F 2017 Xception: deep learning with depthwise separable convolutions *Proc. Conf. on Computer Vision and Pattern Recognition* pp 1251–8
[540] Everingham M *et al* 2015 The PASCAL visual object classes challenge: a retrospective *Int. J. Comput. Vis.* **111** 98–136
[541] Goyal P *et al* 2017 Accurate, large minibatch SGD: training imagenet in 1 hour (arXiv:1706.02677)
[542] Laanait N *et al* 2019 Exascale deep learning for scientific inverse problems (arXiv:1909.11150)
[543] Castelvecchi D 2018 Google unveils search engine for open data *Nature* **561** 161–3
[544] Noy N 2020 Discovering millions of datasets on the web The Keyword (available at: https://blog.google/products/search/discovering-millions-datasets-web)
[545] Plesa N 2020 Machine learning datasets: a list of the biggest machine learning datasets from across the web (available at: www.datasetlist.com)
[546] Dua D and Graff C 2020 UCI machine learning repository (available at: http://archive.ics.uci.edu/ml)
[547] Kaggle Datasets 2020 (available at: www.kaggle.com/datasets)
[548] VisualData 2020 (available at: www.visualdata.io/discovery)
[549] Vanschoren J, Van Rijn J N, Bischl B and Torgo L 2014 OpenML: networked science in machine learning *ACM SIGKDD Explorations Newsl.* **15** 49–60
[550] Stanford S 2020 The best public datasets for machine learning and data science towards AI (available at: https://towardsai.net/datasets)
[551] Datasets for Data Science and Machine Learning 2020 Elite data science (available at: https://elitedatascience.com/datasets)
[552] Iderhoff N 2020 Natural language processing datasets (available at: https://github.com/niderhoff/nlp-datasets)
[553] Deep Learning Datasets 2017 (available at: http://deeplearning.net/datasets)

[554] Hughes I and Hase T 2010 *Measurements and Their Uncertainties: A Practical Guide to Modern Error Analysis* (Oxford: Oxford University Press)
[555] Working Group 1 of the Joint Committee for Guides in Metrology 2008 JCGM 100: 2008 evaluation of measurement data—guide to the expression of uncertainty in measurement Int. Bureau of Weights and Measures (available at: www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf)
[556] Vaux D L, Fidler F and Cumming G 2012 Replicates and repeats - what is the difference and is it significant? A brief discussion of statistics and experimental design *EMBO Rep.* **13** 291–6
[557] Urbach P 1981 On the utility of repeating the 'same' experiment *Australas. J. Philos.* **59** 151–62
[558] Musgrave A 1975 Popper and 'diminishing returns from repeated tests' *Australas. J. Philos.* **53** 248–53
[559] Senior A W *et al* 2020 Improved protein structure prediction using potentials from deep learning *Nature* **577** 706–10
[560] Voß H, Heck C A, Schallmey M and Schallmey A 2020 Database mining for novel bacterial β-etherases, glutathione-dependent lignin-degrading enzymes *Appl. Environ. Microbiol.* **86**
[561] Papers With Code State-of-the-Art Leaderboards 2020 (available at: https://paperswithcode.com/sota)
[562] Krizhevsky A, Nair V and Hinton G 2014 The CIFAR-10 dataset (available at: www.cs.toronto.edu/ kriz/cifar.html)
[563] Krizhevsky A and Hinton G 2009 Learning multiple layers of features from tiny images *Technical Report* Citeseer
[564] LeCun Y, Cortes C and Burges C 2010 MNIST handwritten digit database AT&T Labs (available at: http://yann.lecun.com/exdb/mnist)
[565] Russakovsky O *et al* 2015 ImageNet large scale visual recognition challenge *Int. J. Comput. Vis.* **115** 211–52
[566] Open Access Directory Data Repositories 2020 (available at: http://oad.simmons.edu/oadwiki/Data_repositories)
[567] Nature Scientific Data Rrecommended Data Repositories 2020 (available at: www.nature.com/sdata/policies/repositories)
[568] Zenodo 2020 (available at: https://about.zenodo.org)
[569] Zenodo Frequently Asked Questions 2020 (available at: https://help.zenodo.org)
[570] Ortega D R *et al* 2019 ETDB-Caltech: a blockchain-based distributed public database for electron tomography *PLoS One* **14** e0215531
[571] EMDataResource: Unified Data Resource for 3DEM 2020 (available at: www.emdataresource.org/index.html
[572] Lawson C L *et al* 2016 EMDataBank unified data resource for 3DEM *Nucleic Acids Res.* **44** D396–403
[573] Esquivel-Rodríguez J *et al* 2015 Navigating 3D electron microscopy maps with EM-SURFER *BMC Bioinform.* **16** 181
[574] Lawson C L *et al* 2010 EMDataBank.org: unified data resource for cryoEM *Nucleic Acids Res.* **39** D456–464
[575] Henrick K, Newman R, Tagari M and Chagoyen M 2003 EMDep: a web-based system for the deposition and validation of high-resolution electron microscopy macromolecular structural information *J. Struct. Biol.* **144** 228–37
[576] Tagari M, Newman R, Chagoyen M, Carazo J-M and Henrick K 2002 New electron microscopy database and deposition system *Trends Biochem. Sci.* **27** 589
[577] Iudin A, Korir P K, Salavert-Torres J, Kleywegt G J and Patwardhan A 2016 EMPIAR: a public archive for raw electron microscopy image data *Nat. Methods* **13** 387
[578] Aversa R, Modarres M H, Cozzini S, Ciancio R and Chiusole A 2018 The first annotated set of scanning electron microscopy images for nanoscience *Sci. Data* **5** 180172
[579] Levin B D *et al* 2016 Nanomaterial datasets to advance tomography in scanning transmission electron microscopy *Sci. Data* **3** 1–11
[580] Cerius^2 Modeling Environment: File Formats 2020 (available at: www.chem.cmu.edu/courses/09-560/docs/msi/modenv/D_Files.html)
[581] CrystalMaker: File Formats Supported 2020 (available at: www.crystalmaker.com/support/advice/index.html?topic=cm-file-formats)
[582] Bernstein H J *et al* 2016 Specification of the crystallographic information file format, version 2.0 *J. Appl. Crystallogr.* **49** 277–84
[583] Hall S R and McMahon B 2016 The implementation and evolution of STAR/CIF ontologies: interoperability and preservation of structured data *Data Sci. J.* **15** 3
[584] Brown I D and McMahon B 2002 CIF: the computer language of crystallography *Acta Crystallogr.* B **58** 317–24
[585] Hall S R, Allen F H and Brown I D 1991 The crystallographic information file (CIF): a new standard archive file for crystallography *Acta Crystallogr.* A **47** 655–85
[586] Bruno I *et al* 2017 Crystallography and databases *Data Sci. J.* **16** 38
[587] Crystallographic Databases and Related Resources 2020 (available at: www.iucr.org/resources/data/databases)
[588] Crystal Structure Databases 2020 (available at: https://serc.carleton.edu/research_education/crystallography/xldatabases.html
[589] Quirós M, Gražulis S, Girdzijauskaitė S, Merkys A and Vaitkus A 2018 Using SMILES strings for the description of chemical connectivity in the crystallography open database *J. Cheminf.* **10** 23
[590] Merkys A *et al* 2016 COD::CIF::Parser:an error-correcting CIF parser for the Perl language *J. Appl. Crystallogr.* **49** 292–301
[591] Gražulis S, Merkys A, Vaitkus A and Okulič-Kazarinas M 2015 Computing stoichiometric molecular composition from crystal structures *J. Appl. Crystallogr.* **48** 85–91
[592] Gražulis S *et al* 2012 Crystallography open database (COD): an open-access collection of crystal structures and platform for world-wide collaboration *Nucleic Acids Res.* **40** D420–427
[593] Gražulis S *et al* 2009 Crystallography open database—an open-access collection of crystal structures *J. Appl. Crystallogr.* **42** 726–9
[594] Downs R T and Hall-Wallace M 2003 The American Mineralogist crystal structure database *Am. Mineral.* **88** 247–50
[595] Zagorac D, Müller H, Ruehl S, Zagorac J and Rehme S 2019 Recent developments in the inorganic crystal structure database: theoretical crystal structure data and related features *J. Appl. Crystallogr.* **52** 918–25
[596] Allmann R and Hinek R 2007 The introduction of structure types into the inorganic crystal structure database ICSD *Acta Crystallogr.* A **63** 412–17
[597] Hellenbrandt M 2004 The inorganic crystal structure database (ICSD) - present and future *Crystallogr. Rev.* **10** 17–22
[598] Belsky A, Hellenbrandt M, Karen V L and Luksch P 2002 New developments in the inorganic crystal structure database (ICSD): accessibility in support of materials research and design *Acta Crystallogr.* B **58** 364–9
[599] Bergerhoff G, Brown I and Allen F *et al* 1987 Crystallographic databases *Int. Union Crystallogr.* **360** 77–95
[600] Mighell A D and Karen V L 1996 NIST crystallographic databases for research and analysis *J. Res. Natl Inst. Stand. Technol.* **101** 273
[601] NIST Standard Reference Database 3 2020 (available at: www.nist.gov/srd/nist-standard-reference-database-3)
[602] Kay W *et al* 2017 The kinetics human action video dataset (arXiv:1705.06950)

[603] Abu-El-Haija S *et al* 2016 YouTube-8M: a large-scale video classification benchmark (arXiv:1609.08675)
[604] Rehm G *et al* 2020 QURATOR: innovative technologies for content and data curation (arXiv:2004.12195)
[605] van der Voort S R, Smits M and Klein S 2020 DeepDicomSort: an automatic sorting algorithm for brain magnetic resonance imaging data *Neuroinformatics*
[606] Pezoulas V C *et al* 2019 Medical data quality assessment: on the development of an automated framework for medical data curation *Comput. Biol. Med.* **107** 270–83
[607] Bhat M *et al* 2019 ADeX: a tool for automatic curation of design decision knowledge for architectural decision recommendations *2019 IEEE Int. Conf. on Software Architecture Companion (ICSA-C)* (IEEE) pp 158–61
[608] Thirumuruganathan S, Tang N, Ouzzani M and Doan A 2018 Data curation with deep learning [vision] (arXiv:1803.01384)
[609] Lee K *et al* 2018 Scaling up data curation using deep learning: an application to literature triage in genomic variation resources *PLoS Comput. Biol.* **14** e1006390
[610] Freitas A and Curry E 2016 Big data curation *New Horizons for a Data-Driven Economy* (Berlin: Springer) pp 87–118
[611] European Microcredit Whitepaper 2019 (available at: www.european-microfinance.org/sites/default/files/document/file/paris_europlace_whitepaper_on_microfinance_july_2019.pdf)
[612] Di Cosmo R and Zacchiroli S 2017 Software heritage: why and how to preserve software source code *Proc. 14th Int. Conf. on Digital Preservation (IPRES2017)*
[613] Apache Allura 2020 (available at: https://allura.apache.org)
[614] AWS CodeCommit 2020 (available at: https://aws.amazon.com/codecommit)
[615] Beanstalk 2020 (available at: https://beanstalkapp.com)
[616] BitBucket 2020 (available at: https://bitbucket.org/product)
[617] GitHub 2020 (available at: https://github.com)
[618] GitLab 2020 (available at: https://about.gitlab.com)
[619] Gogs 2020 (available at: https://gogs.io)
[620] Google Cloud Source Repositories 2020 (available at: https://cloud.google.com/source-repositories)
[621] Launchpad 2020 (available at: https://launchpad.net)
[622] Phabricator 2020 (available at: www.phacility.com/phabricator)
[623] Savannah 2020 (available at: https://savannah.gnu.org)
[624] SourceForge 2020 (available at: https://sourceforge.net)
[625] Sheoran J, Blincoe K, Kalliamvakou E, Damian D and Ell J 2014 Understanding watchers on GitHub *Proc. 11th Conf. on Mining Software Repositories* pp 336–9
[626] Vale G, Schmid A, Santos A R, De Almeida E S and Apel S 2020 On the relation between github communication activity and merge conflicts *Empir. Softw. Eng.* **25** 402–33
[627] Bao L, Xia X, Lo D and Murphy G C 2019 A large scale study of long-time contributor prediction for github projects *IEEE Trans. Softw. Eng.* (https://doi.org/10.1109/TSE.2019.2918536)
[628] Elazhary O, Storey M-A, Ernst N and Zaidman A 2019 Do as I do, not as I say: do contribution guidelines match the GitHub contribution process? *2019 IEEE Int. Conf. on Software Maintenance and Evolution (ICSME)* (IEEE) pp 286–90
[629] Pinto G, Steinmacher I and Gerosa M A 2016 More common than tou think: an in-depth study of casual contributors *2016 IEEE 23rd Int. Conf. on Software Analysis, Evolution and Reengineering (SANER)* vol 1 (IEEE) pp 112–23
[630] Kobayakawa N and Yoshida K 2017 How GitHub contributing.md contributes to contributors *2017 IEEE 41st Annual Computer Software and Conf. (COMPSAC)* vol 1 (IEEE) pp 694–6
[631] Lu Y *et al* 2019 Studying in the 'bazaar': an exploratory study of crowdsourced learning in GitHub *IEEE Access* **7** 58930–44
[632] Qiu H S, Li Y L, Padala S, Sarma A and Vasilescu B 2019 The signals that potential contributors look for when choosing open-source projects *Proc. ACM Hum.-Comput. Interact.* **3** 1–29
[633] Alamer G and Alyahya S 2017 Open source software hosting platforms: a collaborative perspective's review *J. Softw.* **12** 274–91
[634] Wikipedia Contributors] 2020 Comparison of source-code-hosting facilities—wikipedia, the free encyclopedia (available at: https://en.wikipedia.org/w/index.php?title=Comparison_of_source-code-hosting_facilities&oldid=964020832) (Accessed 25 June 2020)
[635] Apache Allura Feature Comparison 2020 (available at: https://forge-allura.apache.org/p/allura/wiki/Feature%20Comparison)
[636] Alexa Top Sites 2020 (available at: www.alexa.com/topsites)
[637] How are Alexa's Traffic Rankings Determined 2020 (available at: https://support.alexa.com/hc/en-us/articles/200449744-How-are-Alexa-s-traffic-rankings-determined-)
[638] Haider J and Sundin O 2019 *Invisible Search and Online Search Engines: The Ubiquity of Search in Everyday Life* (Routledge)
[639] Vincent N, Johnson I, Sheehan P and Hecht B 2019 Measuring the importance of user-generated content to search engines *Proc. Int. Conf. on Web and Social Media* vol 13 pp 505–16
[640] Jain A 2013 The role and importance of search engine and search engine optimization *Int. J. Emerg. Trends Technol. Comput. Sci.* **2** 99–102
[641] Brin S and Page L 1998 The anatomy of a large-scale hypertextual web search engine *Comput. Netw.* **30** 107–17
[642] Fröbe M, Bittner J P, Potthast M and Hagen M 2020 The effect of content-equivalent near-duplicates on the evaluation of search engines *Conf. on Information Retrieval* (Springer) pp 12–19
[643] Kostagiolas P, Strzelecki A, Banou C and Lavranos C 2020 The impact of google on discovering scholarly information: managing stm publishers' visibility in google *Collection and Curation*
[644] Gul S, Ali S and Hussain A 2020 Retrieval performance of google, yahoo and bing for navigational queries in the field of 'life science and biomedicine *Data Technol. Appl.* **54** 133–50
[645] Shafi S and Ali S 2019 Retrieval performance of select search engines in the field of physical sciences *NISCAIR-CSIR* pp 117–22
[646] Steiner M, Magin M, Stark B and Geiß S 2020 Seek and you shall find? A content analysis on the diversity of five search engines' results on political queries *Inf. Commun. Soc.* 1–25
[647] Wu S, Zhang Z and Xu C 2019 Evaluating the effectiveness of web search engines on results diversification *Inf. Res.* **24** n1
[648] Rahim I, Mushtaq H and Ahmad S *et al* 2019 Evaluation of search engines using advanced search: comparative analysis of yahoo and bing *Libr. Philos. Pract.*
[649] Tazehkandi M Z and Nowkarizi M Evaluating the effectiveness of google, parsijoo, rismoon, and yooz to retrieve Persian documents *Library Hi Tech* 2020
[650] Gusenbauer M 2019 Google scholar to overshadow them all? Comparing the sizes of 12 academic search engines and bibliographic databases *Scientometrics* **118** 177–214

[651] Hook D W, Porter S J and Herzog C 2018 Dimensions: building context for search and evaluation *Front. Res. Metrics Anal.* **3** 23
[652] Bates J, Best P, McQuilkin J and Taylor B 2017 Will web search engines replace bibliographic databases in the systematic identification of research? *J. Acad. Librariansh.* **43** 8–17
[653] Verheggen K *et al* 2020 Anatomy and evolution of database search engines—a central component of mass spectrometry based proteomic workflows *Mass Spectrom. Rev.* **39** 292–306
[654] Li S *et al* 2020 Deep job understanding at linkedin *Proc. 43rd Int. ACM Conf. on Research and Development in Information Retrieval* pp 2145–8
[655] Agazzi A E 2020 Study of the usability of linkedin: a social media platform meant to connect employers and employees (arXiv:2006.03931)
[656] Forrester A, Björk B-C and Tenopir C 2017 New web services that help authors choose journals *Learn. Publ.* **30** 281–7
[657] Kang D M, Lee C C, Lee S and Lee W 2020 Patent prior art search using deep learning language model *Proc. 24th Symp. on Int. Database Engineering and Applications* pp 1–5
[658] Kang M, Lee S and Lee W 2020 Prior art search using multi-modal embedding of patent documents *2020 IEEE Int. Conf. on Big Data and Smart Computing (BigComp)* (IEEE) pp 548–50
[659] Shalaby W and Zadrozny W 2019 Patent retrieval: a literature review *Knowl. Inf. Syst.* **61** 631–60
[660] Khode A and Jambhorkar S 2017 A literature review on patent information retrieval techniques *Indian J. Sci. Technol.* **10** 1–13
[661] Kong X, Shi Y, Yu S, Liu J and Xia F 2019 Academic social networks: modeling, analysis, mining and applications *J. Netw. Comput. Appl.* **132** 86–103
[662] Makri K, Papadas K and Schlegelmilch B B 2019 Global social networking sites and global identity: a three-country study *J. Bus. Res.* (https://doi.org/10.1016/j.jbusres.2019.11.065)
[663] Acquisti A and Fong C 2020 An experiment in hiring discrimination via online social networks *Manage. Sci.* **66** 1005–24
[664] Mustafaraj E, Lurie E and Devine C 2020 The case for voter-centered audits of search engines during political elections *Proc. 2020th Conf. on Fairness, Accountability and Transparency* pp 559–69
[665] Kulshrestha J *et al* 2019 Search bias quantification: investigating political bias in social media and web search *Inf. Retr. J.* **22** 188–227
[666] Puschmann C 2019 Beyond the bubble: assessing the diversity of political search results *Digit. J.* **7** 824–43
[667] Ray L 2020 2020 google search survey: how much do users trust their search results? MOZ (available at: https://moz.com/blog/2020-google-search-survey)
[668] Johnson D M 2019 Lectures, textbooks, academic calendar and administration: an agenda for change *The Uncertain Future of American Public Higher Education* (Berlin: Springer) pp 75–89
[669] Lin H 2019 Teaching and learning without a textbook: undergraduate student perceptions of open educational resources *Int. Rev. Res. Open Distrib. Learn.* **20** 1–18
[670] Stack Overflow 2020 (available at: https://stackoverflow.com/tour)
[671] Wu Y, Wang S, Bezemer C-P and Inoue K 2019 How do developers utilize source code from stack overflow? *Empir. Softw. Eng.* **24** 637–73
[672] Zhang H, Wang S, Chen T-H and Hassan A E 2019 Reading answers on stack overflow: not enough! *IEEE Trans. Softw. Eng.*
[673] Zhang T, Gao C, Ma L, Lyu M and Kim M 2019 An empirical study of common challenges in developing deep learning applications *2019 IEEE 30th Int. Symp. on Software Reliability Engineering (ISSRE)* (IEEE) pp 104–15
[674] Ragkhitwetsagul C, Krinke J, Paixao M, Bianco G and Oliveto R 2019 Toxic code snippets on stack overflow *IEEE Trans. Softw. Eng.* (https://doi.org/10.1109/TSE.2019.2900307)
[675] Zhang T, Upadhyaya G, Reinhardt A, Rajan H and Kim M 2018 Are code examples on an online Q and A forum reliable?: a study of API misuse on stack overflow *2018 IEEE/ACM 40th Int. Conf. on Software Engineering (ICSE)* (IEEE) pp 886–96
[676] Medium 2020 (available at: https://medium.com)
[677] Machine Learning Subreddit Reddit 2020 (available at: www.reddit.com/r/MachineLearning)
[678] Learn Machine Learning Subreddit Reddit 2020 (available at: www.reddit.com/r/learnmachinelearning)
[679] Mitchell D R G and Schaffer B 2005 Scripting-customised microscopy tools for digital micrograph *Ultramicroscopy* **103** 319–32
[680] DigitalMicrograph Scripts 2020 (available at: www.dmscripting.com/scripts.html)
[681] Internet Archive 2020 (available at: ttps://archive.org)
[682] Kanhabua N *et al* 2016 How to search the internet archive without indexing it *Int. Conf. on Theory and Practice of Digital Libraries* (Springer) pp 147–60
[683] Internet Archive Wayback Machine 2020 (available at: https://archive.org/web)
[684] Bowyer S 2020 The wayback machine: notes on a re-enchantment *Arch. Sci.*
[685] Grotke A 2011 Web archiving at the library of congress *Comput. Libr.* **31** 15–9
[686] About Distill 2020 (available at: https://distill.pub/about)
[687] Lewinson E 2020 My 10 favorite resources for learning data science online Towards Data Science (available at: https://towardsdatascience.com/my-10-favorite-resources-for-learning-data-science-online-c645aa3d0afb)
[688] Chadha H S 2020 Handpicked resources for learning deep learning in 2020 Towards Data Science (available at: https://towardsdatascience.com/handpicked-resources-for-learning-deep-learning-in-2020-e50c6768ab6e)
[689] Besbes A 2020 Here are my top resources to learn deep learning Towards Data Science (available at: https://medium.com/datadriveninvestor/my-top-resources-to-learn-deep-learning-a14d1fc8e95a)
[690] Hutson M 2018 Artificial intelligence faces reproducibility crisis
[691] Baker M 2016 Reproducibility crisis? *Nature* **533** 353–66
[692] Sethi A, Sankaran A, Panwar N, Khare S and Mani S 2017 DLPaper2Code: auto-generation of code from deep learning research papers (arXiv:1711.03543)
[693] Global State of Peer Review Publons 2018 (available at: https://publons.com/static/Publons-Global-State-Of-Peer-Review-2018.pdf)
[694] Tennant J P 2018 The state of the art in peer review *FEMS Microbiol. Lett.* **365**
[695] Walker R and Rocha da Silva P 2015 Emerging trends in peer review—a survey *Front. Neurosci.* **9** 169
[696] Vesper I 2018 Peer reviewers unmasked: largest global survey reveals trends *Nature* (https://doi.org/10.1038/d41586-018-06602-y)
[697] Tan Z-Y, Cai N, Zhou J and Zhang S-G 2019 On performance of peer review for academic journals: analysis based on distributed parallel system *IEEE Access* **7** 19024–32

[698] Kim L, Portenoy J H, West J D and Stovel K W 2019 Scientific journals still matter in the era of academic search engines and preprint archives *J. Assoc. Inf. Sci. Technol.* **71**
[699] Rallison S 2015 What are journals for? *Ann. R. Coll. Surg. Engl.* **97** 89–91
[700] Bornmann L and Mutz R 2015 Growth rates of modern science: a bibliometric analysis based on the number of publications and cited references *J. Assoc. Inf. Sci. Technol.* **66** 2215–22
[701] Kaldas M, Michael S, Hanna J and Yousef G M 2020 Journal impact factor: a bumpy ride in an open space *J. Investigative Med.* **68** 83–7
[702] Orbay K, Miranda R and Orbay M 2020 Building journal impact factor quartile into the assessment of academic performance: a case study *Participatory Educ. Res.* **7** 1–13
[703] Lei L and Sun Y 2020 Should highly cited items be excluded in impact factor calculation? The effect of review articles on journal impact factor *Scientometrics* **122** 1697–706
[704] Top Most Research Tools for Selecting the Best Journal for Your Research Article 2019 Pubrica (available at: https://pubrica.com/academy/2019/11/14/topmost-research-tools-for-selecting-the-best-journal-for-your-research-article)
[705] Hoy M B 2020 Rise of the rxivs: how preprint servers are changing the publishing process *Med. Ref. Serv. Q.* **39** 84–9
[706] Fry N K, Marshall H and Mellins-Cohen T 2019 In praise of preprints *Microb. Genom.* **5**
[707] Rodríguez E G 2019 Preprints and preprint servers as academic communication tools *Revista Cubana de Información en Ciencias de la Salud* **30** 7
[708] About arXiv 2020 (available at: https://arxiv.org/about)
[709] Ginsparg P 2011 ArXiv at 20 *Nature* **476** 145–7
[710] Fraser N, Momeni F, Mayr P and Peters I 2020 The Relationship between biorxiv preprints, citations and altmetrics *Quant. Sci. Stud.* **1** 618–38
[711] Wang Z, Glänzel W and Chen Y 2020 The impact of preprints in library and information science: an analysis of citations, usage and social attention indicators *Scientometrics* **125** 1403–23
[712] Furnival A C and Hubbard B 2020 Open access to scholarly communications: advantages, policy and advocacy *Acceso Abierto a la Información en las Bibliotecas AcadéMicas de AméRica Latina y el Caribe* pp 101–20
[713] Fu D Y and Hughey J J 2019 Meta-research: releasing a preprint is associated with more attention and citations for the peer-reviewed article *Elife* **8** e52646
[714] Niyazov Y *et al* 2016 Open access meets discoverability: citations to articles posted to academia.edu *PLoS One* **11** e0148257
[715] Robinson-Garcia N, Costas R and van Leeuwen T N 2020 State of open access penetration in universities worldwide (arXiv:2003.12273)
[716] Siler K and Frenken K 2020 The pricing of open access journals: diverse niches and sources of value in academic publishing *Quant. Sci. Stud.* **1** 28–59
[717] Green T 2019 Is open access affordable? Why current models do not work and why we need internet-era transformation of scholarly communications *Learn. Publ.* **32** 13–25
[718] Gadd E, Fry J and Creaser C 2018 The influence of journal publisher characteristics on open access policy trends *Scientometrics* **115** 1371–93
[719] Why Should You Publish in Machine Learning: Science and Technology? 2020 IOP Science (available at: https://iopscience.iop.org/journal/2632-2153/page/about-the-journal)
[720] Gibney E 2016 Open journals that piggyback on arXiv gather momentum *Nature News* **530** 117
[721] Martínez-López J I, Barrón-González S and Martínez López A 2019 Which are the tools available for scholars? A review of assisting software for authors during peer reviewing process *Publications* **7** 59
[722] Microsoft Word 2020 (available at: www.microsoft.com/en-gb/microsoft-365/word)
[723] 10 Free MS Word Alternatives You Can Use Today 2020 Investintech (available at: www.investintech.com/resources/articles/tenwordalternatives)
[724] Pignalberi G and Dominici M 2019 Introduction to LATEX and to some of its tools *ArsTEXnica* **28** 8–46
[725] Bransen M and Schulpen G 2018 Pimp your thesis: a minimal introduction to LATEX IC/TC, U.S.S. Proton (available at: https://ussproton.nl/files/careerweeks/20180320-pimpyourthesis.pdf)
[726] Lamport L 1994 *Latex: A Document Preparation System: User's Guide and Reference Manual* (Reading, MA: Addison-Wesley)
[727] Matthews D 2019 Craft beautiful equations in word with LaTeX
[728] Knauff M and Nejasmic J 2014 An efficiency comparison of document preparation systems used in academic research and development *PLoS One* **9** e115069
[729] Why I Write with LaTeX (and Why You Should Too) 2017 Medium (available at: https://medium.com/@marko_kovic/why-i-write-with-latex-and-why-you-should-too-ba6a764fadf9)
[730] Allington D 2016 The LaTeX fetish (or: don't write in LaTeX! It's just for typesetting) (available at: www.danielallington.net/2016/09/the-latex-fetish)
[731] Overleaf Documentation 2020 (available at: www.overleaf.com/learn)
[732] Venkateshaiah A *et al* 2020 Microscopic techniques for the analysis of micro and nanostructures of biopolymers and their derivatives *Polymers* **12** 512
[733] Alqaheem Y and Alomair A A 2020 Microscopy and spectroscopy techniques for characterization of polymeric membranes *Membranes* **10** 33
[734] Morrison K 2019 *Characterisation Methods in Solid State and Materials Science* (Bristol: IOP Publishing)
[735] Maghsoudy-Louyeh S, Kropf M and Tittmann B 2018 Review of progress in atomic force microscopy *Open Neuroimaging J.* **12** 86–104
[736] Rugar D and Hansma P 1990 Atomic force microscopy *Phys. Today* **43** 23–30
[737] Krull A, Hirsch P, Rother C, Schiffrin A and Krull C 2020 Artificial-intelligence-driven scanning probe microscopy *Commun. Phys.* **3** 1–8
[738] Dutta A 2017 Fourier transform infrared spectroscopy *Spectroscopic Methods for Nanomaterials Characterization* (Amsterdam: Elsevier) pp 73–93
[739] Griffiths P R and De Haseth J A 2007 *Fourier Transform Infrared Spectrometry* vol 171 (New York: Wiley)
[740] Chien P-H, Griffith K J, Liu H, Gan Z and Hu Y-Y 2020 Recent advances in solid-state nuclear magnetic resonance techniques for materials research *Ann. Rev. Mater. Res.* **50** 493–520
[741] Lambert J B, Mazzola E P and Ridge C D 2019 *Nuclear Magnetic Resonance Spectroscopy: An Introduction to Principles, Applications and Experimental Methods* (New York: Wiley)

[742] Mlynárik V 2017 Introduction to nuclear magnetic resonance *Anal. Biochem.* **529** 4–9
[743] Rabi I I, Zacharias J R, Millman S and Kusch P 1938 A new method of measuring nuclear magnetic moment *Phys. Rev.* **53** 318
[744] Smith E and Dent G 2019 *Modern Raman Spectroscopy: A Practical Approach* (New York: Wiley)
[745] Jones R R, Hooper D C, Zhang L, Wolverson D and Valev V K 2019 Raman techniques: Fundamentals and frontiers *Nanoscale Res. Lett.* **14** 1–34
[746] Ameh E 2019 A review of basic crystallography and x-ray diffraction applications *Int. J. Adv. Manuf. Technol.* **105** 3289–302
[747] Rostron P, Gaber S and Gaber D 2016 Raman spectroscopy, review *Int. J. Eng. Tech. Res.* **6** 2454–4698
[748] Zhang X, Tan Q-H, Wu J-B, Shi W and Tan P-H 2016 Review on the raman spectroscopy of different types of layered materials *Nanoscale* **8** 6435–50
[749] Epp J 2016 X-ray diffraction (XRD) techniques for materials characterization *Materials Characterization Using Nondestructive Evaluation (NDE) Methods* (Amsterdam: Elsevier) pp 81–124
[750] Keren S *et al* 2008 Noninvasive molecular imaging of small living subjects using raman spectroscopy *Proc. Natl Acad. Sci.* **105** 5844–9
[751] Khan H *et al* 2020 Experimental methods in chemical engineering: x-ray diffraction spectroscopy—XRD *Can. J. Chem. Eng.* **98** 1255–66
[752] Scarborough N M *et al* 2017 Dynamic x-ray diffraction sampling for protein crystal positioning *J. Synchrotron Radiat.* **24** 188–95
[753] Leani J J, Robledo J I and Sánchez H J 2019 Energy dispersive inelastic x-ray scattering spectroscopy—a review *Spectrochim. Acta* B **154** 10–24
[754] Vanhoof C, Bacon J R, Fittschen U E and Vincze L 2020 2020 Atomic spectrometry update—a review of advances in x-ray fluorescence spectrometry and its special applications *J. Anal. At. Spectrom.* **35** 1704–19
[755] Shackley M S 2018 X-ray fluorescence spectrometry (XRF) *Encyclopedia Archaeolog. Sci.* 1–5
[756] Greczynski G and Hultman L 2020 X-ray photoelectron spectroscopy: towards reliable binding energy referencing *Prog. Mater. Sci.* **107** 100591
[757] Baer D R *et al* 2019 Practical guides for x-ray photoelectron spectroscopy: first steps in planning, conducting and reporting XPS measurements *J. Vac. Sci. Technol.* A **37** 031401
[758] Du M and Jacobsen C 2020 Relative merits and limiting factors for x-ray and electron microscopy of thick, hydrated organic materials (revised)
[759] Hsu T 1992 Technique of reflection electron microscopy *Microsc. Res. Tech.* **20** 318–32
[760] Yagi K 1987 Reflection electron microscopy *J. Appl. Crystallogr.* **20** 147–60
[761] Mohammed A and Abdullah A 2018 Scanning electron microscopy (SEM): a review *Proc. 2018 Int. Conf. on Hydraulics and Pneumatics, Băile Govora, Romania* pp 7–9
[762] Goldstein J I *et al* 2017 *Scanning Electron Microscopy and X-ray Microanalysis* (Berlin: Springer)
[763] Keyse R 2018 *Introduction to Scanning Transmission Electron Microscopy* (Routledge)
[764] Pennycook S J and Nellist P D 2011 *Scanning Transmission Electron Microscopy: Imaging and Analysis* (Springer Science and Business Media)
[765] Sutter P 2019 Scanning tunneling microscopy in surface science *Springer Handbook of Microscopy* (Berlin: Springer) p 2
[766] Voigtländer B *et al* 2018 Invited review article: multi-tip scanning tunneling microscopy: experimental techniques and data analysis *Rev. Sci. Instrum.* **89** 101101
[767] Carter C B and Williams D B 2016 *Transmission Electron Microscopy: Diffraction, Imaging and Spectrometry* (Berlin: Springer)
[768] Tang C and Yang Z 2017 Transmission electron microscopy (TEM) *Membrane Characterization* (Amsterdam: Elsevier) pp 145–59
[769] Harris J R 2015 Transmission electron microscopy in molecular structural biology: a historical survey *Arch. Biochem. Biophys.* **581** 3–18
[770] Herzog C, Hook D and Konkiel S 2020 Dimensions: bringing down barriers between scientometricians and data *Quant. Sci. Stud.* **1** 387–95
[771] Bode C, Herzog C, Hook D and McGrath R 2018 A guide to the dimensions data approach *Digit. Sci.*
[772] Adams J *et al* 2018 Dimensions-a collaborative approach to enhancing research discovery *Digit. Sci.*
[773] Gleichmann N SEM vs TEM 2020 Technology networks: analysis and separations (available at: www.technologynetworks.com/analysis/articles/sem-vs-tem-331262)
[774] Owen G 2018 Purchasing an electron microscope?—considerations and scientific strategies to help in the decision making process *Microscopy*
[775] Electron Microscopy Suite: Price List 2020 The Open University (available at: http://www9.open.ac.uk/emsuite/services/price-list)
[776] Electron Microscopy Research Services: Prices Newcastle University (available at: www.ncl.ac.uk/emrs/prices 2020
[777] Sahlgrenska Academy: Prices for Electron Microscopy 2020 University of Gothenburg (available at: https://cf.gu.se/english/centre_for_cellular_imaging/User_Information/Prices/electron-microscopy)
[778] Electron Microscopy: Pricelist 2020 Harvard Medical School (available at: https://electron-microscopy.hms.harvard.edu/pricelist)
[779] Cambridge Advanced Imaging Centre: Services and Charges 2020 University of Cambridge (available at: https://caic.bio.cam.ac.uk/booking/services)
[780] Ichimiya A, Cohen P I and Cohen P I 2004 *Reflection High-Energy Electron Diffraction* (Cambridge: Cambridge University Press)
[781] Braun W 1999 *Applied Rheed: Reflection High-Energy Electron Diffraction During Crystal Growth* vol 154 (Springer Science and Business Media)
[782] Xiang Y, Guo F, Lu T and Wang G 2016 Reflection high-energy electron diffraction measurements of reciprocal space structure of 2D materials *Nanotechnology* **27** 485703
[783] Mašek K, Moroz V and Matolín V 2003 Reflection high-energy electron loss spectroscopy (RHEELS): a new approach in the investigation of epitaxial thin film growth by reflection high-energy electron diffraction (RHEED) *Vacuum* **71** 59–64
[784] Atwater H A and Ahn C C 1991 Reflection electron energy loss spectroscopy during initial stages of Ge growth on Si by molecular beam epitaxy *Appl. Phys. Lett.* **58** 269–71
[785] Yu L *et al* 2020 Aberration corrected spin polarized low energy electron microscope *Ultramicroscopy* **216** 113017
[786] Bauer E 2019 LEEM, SPLEEM and SPELEEM *Springer Handbook of Microscopy* (Berlin: Springer) p 2
[787] Li Q *et al* 2017 A study of chiral magnetic stripe domains within an in-plane virtual magnetic field using SPLEEM *APS* **2017** L50–006

[788] Matsui F 2018 Auger electron spectroscopy *Compendium of Surface and Interface Analysis* (Berlin: Springer) pp 39–44
[789] MacDonald N and Waldrop J 1971 Auger electron spectroscopy in the scanning electron microscope: auger electron images *Appl. Phys. Lett.* **19** 315–8
[790] Scimeca M, Bischetti S, Lamsira H K, Bonfiglio R and Bonanno E 2018 Energy dispersive x-ray (EDX) microanalysis: a powerful tool in biomedical research and diagnosis *Eur. J. Histochem.* **62**
[791] Chen Z *et al* 2016 Quantitative atomic resolution elemental mapping via absolute-scale energy dispersive x-ray spectroscopy *Ultramicroscopy* **168** 7–16
[792] Eggert F, Camus P, Schleifer M and Reinauer F 2018 Benefits from bremsstrahlung distribution evaluation to get unknown information from specimen in SEM and TEM *IOP Conf. Ser.: Mater. Sci. Eng.* **304** 012005
[793] Mohr P J, Newell D B and Taylor B N 2016 CODATA recommended values of the fundamental physical constants: 2014 *J. Phys. Chem. Ref. Data* **45** 043102
[794] Romano A and Marasco A 2018 An introduction to special relativity *Classical Mechanics With Mathematica®* (Berlin: Springer) pp 569–97
[795] French A P 2017 *Special Relativity* (Boca Raton, FL: CRC Press)
[796] Rayleigh L 1879 XXXI. Investigations in optics, with special reference to the spectroscope *London, Edinburgh Dublin Phil. Mag. J. Sci.* **8** 261–74
[797] Ram S, Ward E S and Ober R J 2006 Beyond rayleigh's criterion: a resolution measure with application to single-molecule microscopy *Proc. Natl Acad. Sci.* **103** 4457–62
[798] The Rayleigh Criterion 2020 HyperPhysics (available at: http://hyperphysics.phy-astr.gsu.edu/hbase/phyopt/Raylei.html)
[799] Güémez J, Fiolhais M and Fernández L A 2016 The principle of relativity and the de broglie relation *Am. J. Phys.* **84** 443–7
[800] MacKinnon E 1976 De Broglie's thesis: a critical retrospective *Am. J. Phys.* **44** 1047–55
[801] DeBroglie Wavelength 2020 HyperPhysics (available at: http://hyperphysics.phy-astr.gsu.edu/hbase/quantum/debrog2.html#c5)
[802] Glossary of TEM Terms: Wavelength of Electron 2020 JEOL (available at: www.jeol.co.jp/en/words/emterms/search_result.html?keyword=wavelength%20of%20electron)
[803] Mendenhall M H *et al* 2017 High-precision measurement of the x-ray Cu Kα spectrum *J. Phys. B: At. Mol. Opt. Phys.* **50** 115004
[804] Transmission Electron Microscopy vs Scanning Electron Microscopy 2020 ThermoFisher Scientific (available at: www.thermofisher.com/uk/en/home/materials-science/learning-center/applications/sem-tem-difference.html)
[805] Latychevskaia T 2017 Spatial coherence of electron beams from field emitters and its effect on the resolution of imaged objects *Ultramicroscopy* **175** 121–9
[806] Van Dyck D 2011 Persistent misconceptions about incoherence in electron microscopy *Ultramicroscopy* **111** 894–900
[807] Krumeich F 2011 Properties of electrons, their interactions with matter and applications in electron microscopy *Lab. Inorg. Chem.*
[808] Greffet J-J and Nieto-Vesperinas M 1998 Field theory for generalized bidirectional reflectivity: derivation of Helmholtz's reciprocity principle and Kirchhoff's law *J. Opt. Soc. Am.* A **15** 2735–44
[809] Clarke F and Parry D 1985 Helmholtz reciprocity: its validity and application to reflectometry *Light. Res. Technol.* **17** 1–11
[810] Rose H and Kisielowski C F 2005 On the reciprocity of TEM and STEM *Microsc. Microanal.* **11** 2114
[811] Peters J J P 2017 Structure and ferroelectricity at the atomic level in perovskite oxides PhD Thesis University of Warwick
[812] Yakovlev S, Downing K, Wang X and Balsara N 2010 Advantages of HAADF vs. conventional TEM imaging for study of PSS-PMB diblock copolymer systems *Microsc. Microanal.* **16** 1698–9
[813] Voelkl E, Hoyle D, Howe J, Inada H and Yotsuji T 2017 STEM and TEM: disparate magnification definitions and a way out *Microsc. Microanal.* **23** 56–7
[814] Bendersky L A and Gayle F W 2001 Electron diffraction using transmission electron microscopy *J. Res. Natl Inst. Stand. Technol.* **106** 997
[815] Hubert A, Römer R and Beanland R 2019 Structure refinement from 'digital' large angle convergent beam electron diffraction patterns *Ultramicroscopy* **198** 1–9
[816] Beanland R, Thomas P J, Woodward D I, Thomas P A and Roemer R A 2013 Digital electron diffraction—seeing the whole picture *Acta Crystallogr.* A **69** 427–34
[817] Tanaka M 1994 Convergent-beam electron diffraction *Acta Crystallogr.* A **50** 261–86
[818] Hovden R and Muller D A 2020 Electron tomography for functional nanomaterials (arXiv:2006.01652)
[819] Koneti S *et al* 2019 Fast electron tomography: applications to beam sensitive samples and *in situ* TEM or operando environmental TEM studies *Mater. Charact.* **151** 480–95
[820] Song H *et al* 2019 Electron tomography: a unique tool solving intricate hollow nanostructures *Adv. Mater.* **31** 1801564
[821] Chen M *et al* 2019 A complete data processing workflow for Cryo-ET and subtomogram averaging *Nat. Methods* **16** 1161–8
[822] Ercius P, Alaidi O, Rames M J and Ren G 2015 Electron tomography: a three-dimensional analytic tool for hard and soft materials research *Adv. Mater.* **27** 5638–63
[823] Weyland M and Midgley P A 2004 Electron tomography *Mater. Today* **7** 32–40
[824] Wang Z *et al* 2020 A consensus framework of distributed multiple-tilt reconstruction in electron tomography *J. Comput. Biol.* **27** 212–22
[825] Doerr A 2017 Cryo-electron tomography *Nat. Methods* **14** 34–34
[826] Öktem O 2015 Mathematics of electron tomography *Handbook of Mathematical Methods in Imaging* vol 1
[827] Tichelaar W, Hagen W J, Gorelik T E, Xue L and Mahamid J 2020 TEM bright field imaging of thick specimens: nodes in thon ring patterns *Ultramicroscopy* **216** 113023
[828] Fujii T *et al* 2018 Toward quantitative bright field TEM imaging of ultra thin samples *Microsc. Microanal.* **24** 1612–13
[829] Vander Wal R L 1998 Soot precursor carbonization: visualization using LIF and LII and comparison using bright and dark field TEM *Combust. Flame* **112** 607–16
[830] Bals S, Kabius B, Haider M, Radmilovic V and Kisielowski C 2004 Annular dark field imaging in a TEM *Solid State Commun.* **130** 675–80
[831] Yücelen E, Lazić I and Bosch E G 2018 Phase contrast scanning transmission electron microscopy imaging of light and heavy atoms at the limit of contrast and resolution *Sci. Rep.* **8** 1–10
[832] Krajnak M, McGrouther D, Maneuski D, O'Shea V and McVitie S 2016 Pixelated detectors and improved efficiency for magnetic imaging in STEM differential phase contrast *Ultramicroscopy* **165** 42–50
[833] Lazić I, Bosch E G and Lazar S 2016 Phase contrast STEM for thin samples: integrated differential phase contrast *Ultramicroscopy* **160** 265–80

[834] Müller-Caspary K *et al* 2019 Comparison of first moment STEM with conventional differential phase contrast and the dependence on electron dose *Ultramicroscopy* **203** 95–104
[835] Zhou D *et al* 2016 Sample tilt effects on atom column position determination in ABF-STEM imaging *Ultramicroscopy* **160** 110–17
[836] Okunishi E *et al* 2009 Visualization of light elements at ultrahigh resolution by STEM annular bright field microscopy *Microsc. Microanal.* **15** 164–5
[837] Van den Bos K H *et al* 2016 Unscrambling mixed elements using high angle annular dark field scanning transmission electron microscopy *Phys. Rev. Lett.* **116** 246101
[838] McMullan G, Faruqi A R and Henderson R 2016 Direct electron detectors *Methods in Enzymology* vol 579 (Amsterdam: Elsevier) pp 1–17
[839] McMullan G, Chen S, Henderson R and Faruqi A 2009 Detective quantum efficiency of electron area detectors in electron microscopy *Ultramicroscopy* **109** 1126–43
[840] Torruella P *et al* 2018 Clustering analysis strategies for electron energy loss spectroscopy (EELS) *Ultramicroscopy* **185** 42–8
[841] Pomarico E *et al* 2018 Ultrafast electron energy-loss spectroscopy in transmission electron microscopy *MRS Bull.* **43** 497–503
[842] Koguchi M, Tsuneta R, Anan Y and Nakamae K 2016 Analytical electron microscope based on scanning transmission electron microscope with wavelength dispersive x-ray spectroscopy to realize highly sensitive elemental imaging especially for light elements *Meas. Sci. Technol.* **28** 015904
[843] Tanaka M, Takeguchi M and Furuya K 2008 X-ray analysis and mapping by wavelength dispersive x-ray spectroscopy in an electron microscope *Ultramicroscopy* **108** 1427–31
[844] Schwartz A J, Kumar M, Adams B L and Field D P 2009 *Electron Backscatter Diffraction in Materials Science* vol 2 (Berlin: Springer)
[845] Humphreys F 2001 Review grain and subgrain characterisation by electron backscatter diffraction *J. Mater. Sci.* **36** 3833–54
[846] Winkelmann A, Nolze G, Vos M, Salvat-Pujol F and Werner W 2016 Physics-based simulation models for EBSD: advances and challenges *Nanoscale* **12** 15
[847] Wright S I, Nowell M M and Field D P 2011 A review of strain analysis using electron backscatter diffraction *Microsc. Microanal.* **17** 316
[848] Wilkinson A J, Meaden G and Dingley D J 2009 Mapping strains at the nanoscale using electron back scatter diffraction *Superlattices Microstruct.* **45** 285–94
[849] Wilkinson A J, Meaden G and Dingley D J 2006 High-resolution elastic strain measurement from electron backscatter diffraction patterns: new levels of sensitivity *Ultramicroscopy* **106** 307–13
[850] Wisniewski W, Švančárek P, Prnová A, Parchovianský M and Galusek D 2020 Y_2O_3–Al_2O_3 microsphere crystallization analyzed by electron backscatter diffraction (EBSD) *Sci. Rep.* **10** 1–21
[851] Basu I, Chen M, Loeck M, Al-Samman T and Molodov D 2016 Determination of grain boundary mobility during recrystallization by statistical evaluation of electron backscatter diffraction measurements *Mater. Charact.* **117** 99–112
[852] Zou Y *et al* 2009 Dynamic recrystallization in the particle/particle interfacial region of cold-sprayed nickel coating: electron backscatter diffraction characterization *Scr. Mater.* **61** 899–902
[853] Kirkland E J 2006 Image simulation in transmission electron microscopy Cornell University (available at: http://muller.research.engineering.cornell.edu/sites/WEELS/summer06/mtutor.pdf)
[854] Kirkland E J 2016 Computation in electron microscopy *Acta Crystallogr.* A **72** 1–27
[855] Kirkland E J 2010 *Advanced Computing in Electron Microscopy* (Springer Science and Business Media)
[856] Computem Repository 2017 (available at: https://sourceforge.net/projects/computem)
[857] Dyson M A 2014 Advances in computational methods for transmission electron microscopy simulation and image processing *PhD Thesis* University of Warwick
[858] Peters J J P and Dyson M A 2019 clTEM (available at: https://github.com/JJPPeters/clTEM)
[859] cudaEM Repository 2018 (available at: https://github.com/ningustc/cudaEM)
[860] Barthel J 2018 Dr. Probe: a software for high-resolution stem image simulation *Ultramicroscopy* **193** 1–11
[861] Barthel J 2020 Dr. Probe—STEM image simulation software (available at: https://er-c.org/barthel/drprobe)
[862] Singh S, Ram F and De Graef M 2017 EMsoft: open source software for electron diffraction/image simulations *Microsc. Microanal.* **23** 212–13
[863] EMsoft Github Repository 2020 (available at: https://github.com/EMsoft-org/EMsoft)
[864] Stadelmann P 2015 JEMS (available at: https://web.archive.org/web/20151201081003/http://cimewww.epfl.ch/people/stadelmann/jemsWebSite/jems.html)
[865] Zuo J and Spence J 2013 *Electron Microdiffraction* (Springer Science and Business Media)
[866] Lobato I, Van Aert S and Verbeeck J 2016 Accurate and fast electron microscopy simulations using the open source MULTEM program *European Microscopy Congress 2016: Proc.* (Wiley Online Library) pp 531–2
[867] Lobato I, Van Aert S and Verbeeck J 2016 Progress and new advances in simulating electron microscopy datasets using MULTEM *Ultramicroscopy* **168** 17–27
[868] Lobato I and Van Dyck D 2015 MULTEM: a new multislice program to perform accurate and fast electron diffraction and imaging simulations using graphics processing units with CUDA *Ultramicroscopy* **156** 9–17
[869] O'Keefe M A and Kilaas R 1988 Advances in high-resolution image simulation *Conf. Proceeding*
[870] Electron Direct Methods 2020 (available at: www.numis.northwestern.edu/edm)
[871] Northwestern University Multislice and Imaging System 2020 (available at: www.numis.northwestern.edu/Software)
[872] Pryor A, Ophus C and Miao J 2017 A streaming multi-GPU implementation of image simulation algorithms for scanning transmission electron eicroscopy *Adv. Struc. Chem. Imaging* **3** 15
[873] Ophus C 2017 A fast image simulation algorithm for scanning transmission electron microscopy *Adv. Struct. Chem. Imaging* **3** 13
[874] Prismatic Repository 2020 (available at: https://github.com/prism-em/prismatic)
[875] QSTEM 2020 (available at: www.physics.hu-berlin.de/en/sem/software/software_qstem)
[876] Gómez-Rodríguez A, Beltrán-del Río L and Herrera-Becerra R 2010 Simulatem: multislice simulations for general objects *Ultramicroscopy* **110** 95–104
[877] STEM-CELL 2020 (available at: http://tem-s3.nano.cnr.it/?page_id=2)
[878] Tempas 2020 (available at: www.totalresolution.com/)
[879] Ishizuka K 2002 A practical approach for stem image simulation based on the FFT multislice method *Ultramicroscopy* **90** 71–83

[880] Ishizuka K 2001 Prospects of atomic resolution imaging with an aberration-corrected STEM *Microscopy* **50** 291–305
[881] Ishizuka K 1982 Multislice formula for inclined illumination *Acta Crystallogr.* A **38** 773–9
[882] Ishizuka K 1980 Contrast transfer of crystal images in TEM *Ultramicroscopy* **5** 55–65
[883] Ishizuka K and Uyeda N 1977 A new theoretical and practical approach to the multislice method *Acta Crystallogr.* A **33** 740–9
[884] HREM Simulation Suite 2020 HREM Research (available at: www.hremresearch.com/Eng/simulation.html)
[885] Gianola S, Jesus T S, Bargeri S and Castellini G 2020 Publish or perish: reporting characteristics of peer-reviewed publications, pre-prints and registered studies on the COVID-19 pandemic *medRxiv*
[886] Nielsen P and Davison R M 2020 Predatory journals: a sign of an unhealthy publish or perish game? *Inf. Syst. J.* **30** 635–8
[887] Génova G and de la Vara J L 2019 The problem is not professional publishing, but the publish-or-perish culture *Sci. Eng. Ethics* **25** 617–19
[888] Zuo J-M and Weickenmeier A 1995 On the beam selection and convergence in the bloch-wave method *Ultramicroscopy* **57** 375–83
[889] Yang Y, Yang Q, Huang J, Cai C and Lin J 2017 Quantitative comparison between real space and bloch wave methods in image simulation *Micron* **100** 73–8
[890] Peng Y, Nellist P D and Pennycook S J 2004 HAADF-STEM imaging with sub-angstrom probes: a full bloch wave analysis *J. Electron Microsc.* **53** 257–66
[891] Cheng L, Ming Y and Ding Z 2018 Bohmian trajectory-bloch wave approach to dynamical simulation of electron diffraction in crystal *New J. Phys.* **20** 113004
[892] Beanland R *et al* 2020 Felix (available at: https://github.com/RudoRoemer/Felix)
[893] Morimura T and Hasaka M 2009 Bloch-wave-based STEM image simulation with layer-by-layer representation *Ultramicroscopy* **109** 1203–9
[894] Gatan Microscopy Suite Software 2020 (available at: www.gatan.com/products/tem-analysis/gatan-microscopy-suite-software)
[895] FELMI/ZFE Script Database 2020 (available at: www.felmi-zfe.at/dm-script)
[896] Gatan Scripts Library 2020 (available at: www.gatan.com/resources/scripts-library)
[897] Potapov P 2020 TemDM: software for tem in digitalmicrograph (available at: http://temdm.com/web)
[898] Koch C 2016 Electron microscopy software (available at: www.physics.hu-berlin.de/en/sem/software)
[899] Schaffer B 2015 'How to script…'—digital micrograph scripting handbook (available at: http://digitalmicrograph-scripting.tavernmaker.de/HowToScript_index.htm)
[900] Mitchell D 2014 A guide to compiling C++ code to create plugins for DigitalMicrograph (GMS 2.x) Dave Mitchell's DigitalMicrograph Scripting Website (available at: www.dmscripting.com/tutorial_compiling_plugins_for_GMS2.pdf)
[901] Miller B and Mick S 2019 Real-time data processing using python in digitalmicrograph *Microsc. Microanal.* **25** 234–5
[902] Hoffman C 2019 RAM disks explained: what they are and why you probably shouldn't use one How-To Geek (available at: www.howtogeek.com/171432/ram-disks-explained-what-they-are-and-why-you-probably-shouldnt-use-one)
[903] Coughlin T, Hoyt R and Handy J 2017 Digital storage and memory technology (part 1) IEEE Technology Trend Paper (available at: www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-industry-advisory-board/digital-storage-memory-technology.pdf)
[904] A dedicated Site for Quantitative Electron Microscopy 2020 HREM research (available at: www.hremresearch.com/index.html)
[905] Rene de Cotret L P 2019 TCP socket plug-in for gatan microscopy suite 3.x (available at: https://github.com/LaurentRDC/gms-socket-plugin)
[906] Schorb M, Haberbosch I, Hagen W J, Schwab Y and Mastronarde D N 2019 Software tools for automated transmission electron microscopy *Nat. Methods* **16** 471–7
[907] Peters J J P 2018 DM stack builder (available at: https://github.com/JJPPeters/DM-Stack-Builder)
[908] Wolf D, Lubk A and Lichte H 2014 Weighted simultaneous iterative reconstruction technique for single-axis tomography *Ultramicroscopy* **136** 15–25
[909] Wolf D 2013 Tomography menu (available at: http://wwwpub.zih.tu-dresden.de/ dwolf/)
[910] Schindelin J, Rueden C T, Hiner M C and Eliceiri K W 2015 The ImageJ ecosystem: an open platform for biomedical image analysis *Mol. Reprod. Dev.* **82** 518–29
[911] EM Software 2020 EMDataResource (available at: www.emdataresource.org/emsoftware.html)
[912] Software Tools For Molecular Microscopy 2020 WikiBooks (available at: https://en.wikibooks.org/wiki/Software_Tools_For_Molecular_Microscopy)
[913] Centre for Microscopy and Microanalysis: Online Tools: Scientific Freeware 2020 University of Queensland (available at: https://cmm.centre.uq.edu.au/online-tools)
[914] Ben-Nun T and Hoefler T 2019 Demystifying parallel and distributed deep learning: an in-depth concurrency analysis *ACM Comput. Surv. (CSUR)* **52** 1–43
[915] Dryden N *et al* 2019 Channel and filter parallelism for large-scale CNN training *Proc. Int. Conf. for High Performance Computing, Networking, Storage and Analysis* pp 1–20
[916] Nwankpa C, Ijomah W, Gachagan A and Marshall S 2018 Activation functions: comparison of trends in practice and research for deep learning (arXiv:1811.03378)
[917] Hayou S, Doucet A and Rousseau J 2019 On the impact of the activation function on deep neural networks training (arXiv:1902.06853)
[918] Roos M 2019 Deep learning neurons versus biological neurons Towards Data Science (available at: https://towardsdatascience.com/deep-learning-versus-biological-neurons-floating-point-numbers-spikes-and-neurotransmitters-6eebfa3390e9)
[919] Eldan R and Shamir O 2016 The power of depth for feedforward neural networks *Conf. on Learning Theory* pp 907–40
[920] Telgarsky M 2016 Benefits of depth in neural networks (arXiv:1602.04485)
[921] Ba J and Caruana R 2014 Do deep nets really need to be deep? *Advances in Neural Information Processing Systems* pp 2654–62
[922] Lee J *et al* 2019 Wide neural networks of any depth evolve as linear models under gradient descent *Advances in Neural Information Processing Systems* pp 8572–83
[923] Yun C, Sra S and Jadbabaie A 2018 Small nonlinearities in activation functions create bad local minima in neural networks (arXiv:1802.03487)
[924] Nair V and Hinton G E 2010 Rectified linear units improve restricted boltzmann machines *Proc. 27th Int. Conf. on Machine Learning (ICML-10)* pp 807–14
[925] Glorot X, Bordes A and Bengio Y 2011 Deep sparse rectifier neural networks *Proc. 14th Int. Conf. on Artificial Intelligence and Statistics* pp 315–23

[926] Maas A L, Hannun A Y and Ng A Y 2013 Rectifier nonlinearities improve neural network acoustic models *Proc. Int. Conf. on Machine Learning* vol 30 p 3
[927] Chen Y *et al* 2020 Dynamic ReLU (arXiv:2003.10027)
[928] Xu B, Wang N, Chen T and Li M 2015 Empirical evaluation of rectified activations in convolutional network (arXiv:1505.00853)
[929] Pedamonti D 2018 Comparison of non-linear activation functions for deep neural networks on MNIST classification task (arXiv:1804.02763)
[930] Chris 2019 Leaky ReLU: Improving Traditional ReLU MachineCurve (available at: www.machinecurve.com/index.php/2019/10/15/leaky-relu-improving-traditional-relu)
[931] Arnekvist I, Carvalho J F, Kragic D and Stork J A 2020 The effect of target normalization and momentum on dying ReLU (arXiv:2005.06195)
[932] Lu L, Shin Y, Su Y and Karniadakis G E 2019 Dying ReLU and initialization: theory and numerical examples (arXiv:1903.06733)
[933] Douglas S C and Yu J 2018 Why RELU units sometimes die: analysis of single-unit error backpropagation in neural networks *2018 52nd Conf. on Signals, Systems and Computers* (IEEE) pp 864–8
[934] Krizhevsky A and Hinton G 2010 Convolutional deep belief networks on CIFAR-10 *Tech. Rep.* **40** 1–9
[935] Shang W, Sohn K, Almeida D and Lee H 2016 Understanding and improving convolutional neural networks via concatenated rectified linear units *Int. Conf. on Machine Learning* pp 2217–25
[936] Gao H, Cai L and Ji S 2020 Adaptive convolutional ReLUs *AAAI* pp 3914–21
[937] Eidnes L and Nøkland A 2017 Shifting mean activation towards zero with bipolar activation functions (arXiv:1709.04054)
[938] Jiang X, Pang Y, Li X, Pan J and Xie Y 2018 Deep neural networks with elastic rectified linear units for object recognition *Neurocomputing* **275** 1132–9
[939] Basirat M and ROTH P 2020 L∗ ReLU: piece-wise linear activation functions for deep fine-grained visual categorization *The IEEE Conf. on Applications of Computer Vision* pp 1218–27
[940] Clevert D-A, Unterthiner T and Hochreiter S 2015 Fast and accurate deep network learning by exponential linear units (ELUs) (arXiv:1511.07289)
[941] Klambauer G, Unterthiner T, Mayr A and Hochreiter S 2017 Self-normalizing neural networks *Advances in Neural Information Processing Systems* pp 971–80
[942] Hryniowski A and Wong A 2019 DeepLABNet: end-to-end learning of deep radial basis networks with fully learnable basis functions (arXiv:1911.09257)
[943] Dash C S K, Behera A K, Dehuri S and Cho S-B 2016 Radial basis function neural networks: a topical state-of-the-art survey *Open Computer Science* **1** 33–63
[944] Orr M J L 1996 Introduction to radial basis function networks (available at: www.cc.gatech.edu/ isbell/tutorials/rbf-intro.pdf)
[945] Jang J-S and Sun C-T 1993 Functional equivalence between radial basis function networks and fuzzy inference systems *IEEE Trans. Neural Netw.* **4** 156–9
[946] Wuraola A and Patel N 2018 Computationally efficient radial basis function *Int. Conf. on Neural Information Processing* (Springer) pp 103–12
[947] Cervantes J, Garcia-Lamont F, Rodríguez-Mazahua L and Lopez A 2020 A comprehensive survey on support vector machine classification: applications, challenges and trends *Neurocomputing* **408** 189–215
[948] Scholkopf B and Smola A J 2018 *Learning With Kernels: Support Vector Machines, Regularization, Optimization and Beyond* (Adaptive Computation and Machine Learning Series)
[949] Tavara S 2019 Parallel computing of support vector machines: a survey *ACM Computing Surveys (CSUR)* **51** 1–38
[950] Kundu A *et al* 2019 K-TanH: hardware efficient activations for deep learning (arXiv:1909.07729)
[951] LeCun Y A, Bottou L, Orr G B and Müller K-R 2012 Efficient backprop *Neural Networks: Tricks of the Trade* (Berlin: Springer) pp 9–48
[952] Abdelouahab K, Pelcat M and Berry F 2017 Why TanH is a hardware friendly activation function for CNNs *Proc. 11th Int. Conf. on Distributed Smart Cameras* pp 199–201
[953] Gulcehre C, Moczulski M, Denil M and Bengio Y 2016 Noisy activation functions *Int. Conf. on Machine Learning* pp 3059–68
[954] Dunne R A and Campbell N A 1997 On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function *Proc. 8th Conf. on Neural Networks, Melbourne* vol 181 (Citeseer) p 185
[955] Dumoulin V and Visin F 2018 A guide to convolution arithmetic for deep learning (arXiv:1603.07285)
[956] Graham B 2014 Fractional max-pooling (arXiv:1412.6071)
[957] Springenberg J T, Dosovitskiy A, Brox T and Riedmiller M 2014 Striving for simplicity: the all convolutional net (arXiv:1412.6806)
[958] Sabour S, Frosst N and Hinton G E 2017 Dynamic routing between capsules *Advances in Neural Information Processing Systems* pp 3856–66
[959] Luo C *et al* 2018 Cosine normalization: using cosine similarity instead of dot product in neural networks *Int. Conf. on Artificial Neural Networks* (Springer) pp 382–91
[960] Nader A and Azar D 2020 Searching for activation functions using a self-adaptive evolutionary algorithm *Proc. 2020 Genetic and Evolutionary Conf. Companion* pp 145–6
[961] Ramachandran P, Zoph B and Le Q 2018 Searching for activation functions Google Research (avaliable at: https://research.google/pubs/pub46503)
[962] Bingham G and Miikkulainen R 2020 Discovering parametric activation functions (arXiv:2006.03179)
[963] Ertuğrul O F 2018 A novel type of activation function in artificial neural networks: trained activation function *Neural Netw.* **99** 148–57
[964] Lau M M and Lim K H 2018 Review of adaptive activation function in deep neural network *2018 IEEE- Conf. on Biomedical Engineering and Sciences (IECBES)* (IEEE) pp 686–90
[965] Chung H, Lee S J and Park J G 2016 Deep neural network using trainable activation functions *2016 Int. Conf. on Neural Networks (IJCNN)* (IEEE) pp 348–52
[966] Agostinelli F, Hoffman M, Sadowski P and Baldi P 2014 Learning activation functions to improve deep neural networks (arXiv:1412.6830)
[967] Wu Y, Zhao M and Ding X 1997 Beyond weights adaptation: a new neuron model with trainable activation function and its supervised learning *Proc. of Int. Conf. on Neural Networks (ICNN'97)* vol 2 (IEEE) pp 1152–7
[968] Lee J *et al* 2019 ProbAct: a probabilistic activation function for deep neural networks (arXiv:1905.10761)
[969] Kingma D P and Welling M 2014 Auto-encoding variational bayes (arXiv:1312.6114)

[970] Springenberg J T and Riedmiller M 2013 Improving deep neural networks with probabilistic maxout units (arXiv:1312.6116)
[971] Bawa V S and Kumar V 2019 Linearized sigmoidal activation: a novel activation function with tractable non-linear characteristics to boost representation capability *Expert Syst. Appl.* **120** 346–56
[972] Kurita K 2018 An overview of normalization methods in deep learning machine learning explained (avaliable at: https://mlexplained.com/2018/11/30/an-overview-of-normalization-methods-in-deep-learning)
[973] Ren M, Liao R, Urtasun R, Sinz F H and Zemel R S 2016 Normalizing the normalizers: comparing and extending network normalization schemes (arXiv:1611.04520)
[974] Liao Q, Kawaguchi K and Poggio T 2016 Streaming normalization: towards simpler and more biologically-plausible normalizations for online and recurrent learning (arXiv:1610.06160)
[975] Santurkar S, Tsipras D, Ilyas A and Madry A 2018 How does batch normalization help optimization? *Advances in Neural Information Processing Systems* pp 2483–93
[976] Ioffe S and Szegedy C 2015 Batch normalization: accelerating deep network training by reducing internal covariate shift (arXiv:1502.03167)
[977] Bjorck N, Gomes C P, Selman B and Weinberger K Q 2018 Understanding batch normalization *Advances in Neural Information Processing Systems* pp 7694–705
[978] Yang G, Pennington J, Rao V, Sohl-Dickstein J and Schoenholz S S 2019 A mean field theory of batch normalization (arXiv:1902.08129)
[979] Ioffe S and Cortes C 2019 Batch normalization layers US Patent 10,417,562
[980] Lian X and Liu J 2019 Revisit batch normalization: new understanding and refinement via composition optimization *The 22nd Int. Conf. on Artificial Intelligence and Statistics* pp 3254–63
[981] Gao P, Yu L, Wu Y and Li J 2018 Low latency RNN inference with cellular batching *Proc. 13th Conf* pp 1–15
[982] Fang Z, Hong D and Gupta R K 2019 Serving deep neural networks at the cloud edge for vision applications on mobile platforms *Proc. 10th ACM Multimedia Conf.* pp 36–47
[983] Das D *et al* 2016 Distributed deep learning using synchronous stochastic gradient descent (arXiv:1602.06709)
[984] Keskar N S, Mudigere D, Nocedal J, Smelyanskiy M and Tang P T P 2016 On large-batch training for deep learning: generalization gap and sharp minima (arXiv:1602.06709)
[985] Masters D and Luschi C 2018 Revisiting small batch training for deep neural networks (arXiv:1804.07612)
[986] You Y, Gitman I and Ginsburg B 2017 Scaling SGD batch size to 32k for ImageNet training *Technical Report* UCB/EECS-2017-156 (Berkeley, CA: EECS Department, University of California)
[987] Devarakonda A, Naumov M and Garland M 2017 AdaBatch: adaptive batch sizes for training deep neural networks (arXiv:1712.02029)
[988] Hoffer E *et al* 2019 Augment your batch: better training with larger batches (arXiv:1901.09335)
[989] Hasani M and Khotanlou H 2019 An empirical study on position of the batch normalization layer in convolutional neural networks *2019 5th Conf. on Signal Processing and Intelligent Systems (ICSPIS)* (IEEE) pp 1–4
[990] Mishkin D 2016 Batch normalization benchmarks (available at: https://github.com/ducha-aiki/caffenet-benchmark/blob/master/batchnorm.md)
[991] Nado Z *et al* 2020 Evaluating prediction-time batch normalization for robustness under covariate shift (arXiv:2006.10963)
[992] Zha D, Lai K-H, Zhou K and Hu X 2019 Experience replay optimization (arXiv:1906.08387)
[993] Schaul T, Quan J, Antonoglou I and Silver D 2015 Prioritized experience replay (arXiv:1511.05952)
[994] Ioffe S 2017 Batch renormalization: towards reducing minibatch dependence in batch-normalized models *Advances in Neural Information Processing Systems* pp 1945–53
[995] Salimans T *et al* 2016 Improved techniques for training GANs *Advances in Neural Information Processing Systems* pp 2234–42
[996] Chiley V *et al* 2019 Online normalization for training neural networks *Advances in Neural Information Processing Systems* pp 8433–43
[997] Hoffer E, Banner R, Golan I and Soudry D 2018 Norm matters: efficient and accurate normalization schemes in deep networks *Advances in Neural Information Processing Systems* pp 2160–70
[998] Ba J L, Kiros J R and Hinton G E 2016 Layer normalization (arXiv:1607.06450)
[999] Xu J, Sun X, Zhang Z, Zhao G and Lin J 2019 Understanding and improving layer normalization *Advances in Neural Information Processing Systems* pp 4381–91
[1000] Ulyanov D, Vedaldi A and Lempitsky V 2017 Instance normalization: the missing ingredient for fast stylization (arXiv:1607.08022)
[1001] Jing Y *et al* 2019 Neural style transfer: a review *IEEE Trans. Vis. Comput. Graph.* **26** 3365–85
[1002] Gatys L A, Ecker A S and Bethge M 2016 Image style transfer using convolutional neural networks *Proc. Conf. on Computer Vision and Pattern Recognition* pp 2414–23
[1003] Gatys L A, Ecker A S and Bethge M 2015 A neural algorithm of artistic style (arXiv:1508.06576)
[1004] Zhu J-Y, Park T, Isola P and Efros A A 2017 Unpaired image-to-image translation using cycle-consistent adversarial networks *Proc. IEEE Int. Conf. on Computer Vision* pp 2223–32
[1005] Li Y, Wang N, Liu J and Hou X 2017 Demystifying neural style transfer (arXiv:1701.01036)
[1006] Wu Y and He K 2018 Group normalization *Proc. Conf. on Computer Vision (ECCV)* pp 3–19
[1007] Luo P, Peng Z, Ren J and Zhang R2018 Do normalization layers in a deep ConvNet really need to be distinct? (arXiv:1811.07727)
[1008] Luo P, Ren J, Peng Z, Zhang R. and Li J 2018 Differentiable learning-to-normalize via switchable normalization (arXiv:1806.10779)
[1009] Nam H and Kim H-E 2018 Batch-instance normalization for adaptively style-invariant neural networks *Advances in Neural Information Processing Systems* pp 2558–67
[1010] Hao K 2019 We analyzed 16,625 papers to figure out where AI is headed next *MIT Technol. Rev.*
[1011] Cooijmans T, Ballas N, Laurent C, Gülçehre Ç and Courville A 2016 Recurrent batch normalization (arXiv:1603.09025)
[1012] Liao Q and Poggio T 2016 Bridging the gaps between residual learning, recurrent neural networks and visual cortex (arXiv:1604.03640)
[1013] Laurent C, Pereyra G, Brakel P, Zhang Y and Bengio Y 2016 Batch normalized recurrent neural networks *2016 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 2657–61
[1014] Salimans T and Kingma D P 2016 Weight normalization: a simple reparameterization to accelerate training of deep neural networks *Advances in Neural Information Processing Systems* pp 901–9
[1015] Qiao S, Wang H, Liu C, Shen W. and Yuille A 2019 Weight standardization (arXiv:1903.10520)

[1016] Gitman I and Ginsburg B 2017 Comparison of batch normalization and weight normalization algorithms for the large-scale image classification (arXiv:1709.08145)
[1017] Miyato T, Kataoka T, Koyama M and Yoshida Y 2018 Spectral normalization for generative adversarial networks (arXiv:1802.05957)
[1018] Wood G R and Zhang B P 1996 Estimation of the Lipschitz constant of a function *J. Glob. Optim.* **8** 91–103
[1019] Hui J 2019 Machine learning—singular value decomposition (SVD) and principal component analysis (PCA) Medium (available at: https://medium.com/@jonathan_hui/machine-learning-singular-value-decomposition-svd-principal-component-analysis-pca-1d45e885e491)
[1020] Afham M 2020 Singular value decomposition and its applications in principal component analysis Towards Data Science (available at: https://towardsdatascience.com/singular-value-decomposition-and-its-applications-in-principal-component-analysis-5b7a5f08d0bd)
[1021] Wall M E, Rechtsteiner A and Rocha L M 2003 Singular value decomposition and principal component analysis *A Practical Approach to Microarray Data Analysis* (Berlin: Springer) pp 91–109
[1022] Klema V and Laub A 1980 The singular value decomposition: its computation and some applications *IEEE Trans. Autom. Control* **25** 164–76
[1023] Yoshida Y and Miyato T 2017 Spectral norm regularization for improving the generalizability of deep learning (arXiv:1705.10941)
[1024] Golub G H and Van der Vorst H A 2000 Eigenvalue computation in the 20th century *J. Comput. Appl. Math.* **123** 35–65
[1025] Nguyen T Q and Salazar J 2019 Transformers without tears: improving the normalization of self-attention (arXiv:1910.05895)
[1026] Nguyen T Q and Chiang D 2017 Improving lexical choice in neural machine translation (arXiv:1710.01329)
[1027] Stewart M 2019 Simple introduction to convolutional neural networks Towards Data Science (available at: https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac)
[1028] Wu J 2017 Introduction to convolutional neural networks *National Key Lab for Novel Software Technology* **5** 23
[1029] McCann M T, Jin K H and Unser M 2017 Convolutional neural networks for inverse problems in imaging: a review *IEEE Signal Process. Mag.* **34** 85–95
[1030] O'Shea K and Nash R 2015 An introduction to convolutional neural networks (arXiv:1511.08458)
[1031] Hubel D H and Wiesel T N 1968 Receptive fields and functional architecture of monkey striate cortex *J. Physiol.* **195** 215–43
[1032] Fukushima K 1980 A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position *Biol. Cybern.* **36** 193–202
[1033] Fukushima K and Miyake S 1982 Neocognitron: a self-organizing neural network nodel for a mechanism of visual pattern recognition *Competition and Cooperation in Neural Nets* (Berlin: Springer) pp 267–85
[1034] Fukushima K 1988 Neocognitron: a hierarchical neural network capable of visual pattern recognition *Neural Netw.* **1** 119–30
[1035] Fukushima K 2003 Neocognitron for handwritten digit recognition *Neurocomputing* **51** 161–80
[1036] Atlas L E, Homma T and Marks R J I I 1988 An artificial neural network for spatio-temporal bipolar patterns: application to phoneme classification *Neural Information Processing Systems* pp 31–40
[1037] LeCun Y, Bottou L, Bengio Y and Haffner P 1998 Gradient-based learning applied to document recognition *Proc. IEEE* **86** 2278–324
[1038] LeCun Y, Haffner P, Bottou L and Bengio Y 1999 Object recognition with gradient-based learning *Shape, Contour and Grouping in Computer Vision* (Berlin: Springer) pp 319–45
[1039] Cireşan D C, Meier U, Gambardella L M and Schmidhuber J 2010 Deep, big, simple neural nets for handwritten digit recognition *Neural Comput.* **22** 3207–20
[1040] Yao G, Lei T and Zhong J 2019 A review of convolutional-neural-network-based action recognition *Pattern Recognit. Lett.* **118** 14–22
[1041] Gupta A *et al* 2019 Deep learning in image cytometry: a review *Cytometry* A **95** 366–80
[1042] Ma S *et al* 2019 Image and video compression with neural networks: a review *IEEE Trans. Circuits Syst. Video Technol.* **30** 1683–98
[1043] Liu D, Li Y, Lin J, Li H and Wu F 2020 Deep learning-based video coding: a review and a case study *ACM Comput. Surv. (CSUR)* **53** 1–35
[1044] Bouwmans T, Javed S, Sultana M and Jung S K 2019 Deep neural network concepts for background subtraction: a systematic review and comparative evaluation *Neural Netw.* **117** 8–66
[1045] Anwar S M *et al* 2018 Medical image analysis using convolutional neural networks: a review *J. Med. Syst.* **42** 226
[1046] Soffer S *et al* 2019 Convolutional neural networks for radiologic images: a radiologist's guide *Radiology* **290** 590–606
[1047] Yamashita R, Nishio M, Do R K G and Togashi K 2018 Convolutional neural networks: an overview and application in radiology *Insights Imaging* **9** 611–29
[1048] Bernal J *et al* 2019 Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review *Artif. Intell. Med.* **95** 64–81
[1049] Fu Y *et al* 2020 Deep learning in medical image registration: a review *Phys. Med. Biol.* **65** 20TR01
[1050] Badar M, Haris M and Fatima A 2020 Application of deep learning for retinal image analysis: a review *Comput. Sci. Rev.* **35** 100203
[1051] Litjens G *et al* 2017 A survey on deep learning in medical image analysis *Med. Image Anal.* **42** 60–88
[1052] Liu J *et al* 2018 Applications of deep learning to MRI images: a survey *Big Data Min. Analytics* **1** 1–18
[1053] Zhao Z-Q, Zheng P, Xu S-t and Wu X 2019 Object detection with deep learning: a review *IEEE Trans. Neural Networks Learn. Syst.* **30** 3212–32
[1054] Wang W *et al* 2019 Salient object detection in the deep learning era: an in-depth survey (arXiv:1904.09146)
[1055] Minaee S *et al* 2020 Deep learning based text classification: a comprehensive review (arXiv:2004.03705)
[1056] TensorFlow Core v2.2.0 Python Documentation for Convolutional Layer 2020 (available at: https://web.archive.org/web/20200520184050/www.tensorflow.org/api_docs/python/tf/nn/convolution)
[1057] McAndrew A 2015 *A Computational Introduction to Digital Image Processing* (Boca Raton, FL: CRC Press)
[1058] Smoothing Images 2019 OpenCV documentation (available at: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html)
[1059] Vairalkar M K and Nimbhorkar S 2012 Edge detection of images using sobel operator *Int. J. Emerg. Technol. Adv. Eng.* **2** 291–3
[1060] Bogdan V, Bonchiş C and Orhei C 2019 Custom extended sobel filters (arXiv:1910.00138)
[1061] Jähne B, Scharr H and Körkel S *et al* 1999 Principles of filter design *Handbook of Computer Vision and Applications* vol 2 (New York: Academic) pp 125–51

[1062] Scharr H 2000 Optimal operators in digital image processing PhD Thesis University of Heidelberg in German
[1063] Kawalec-Latała E 2014 Edge detection on images of pseudoimpedance section supported by context and adaptive transformation model images *Stud. Geotech. Mech.* **36** 29–36
[1064] Roberts L G 1963 Machine perception of three-dimensional solids PhD Thesis Massachusetts Institute of Technology
[1065] Prewitt J M 1970 Object enhancement and extraction *Picture Processing and Psychopictorics* **10** 15–9
[1066] Jin J, Dundar A and Culurciello E 2014 Flattened convolutional neural networks for feedforward acceleration (arXiv:1412.5474)
[1067] Chen J, Lu Z, Xue J-H and Liao Q 2020 XSepConv: extremely separated convolution (arXiv:2002.12046)
[1068] Jaderberg M, Vedaldi A and Zisserman A 2014 Speeding up convolutional neural networks with low rank expansions (arXiv:1405.3866)
[1069] Wu S, Wang G, Tang P, Chen F and Shi L 2019 Convolution with even-sized kernels and symmetric padding *Advances in Neural Information Processing Systems* pp 1194–205
[1070] Kossaifi J, Bulat A, Panagakis Y, Pantic M and Cambridge S A 2019 Efficient *N*-dimensional convolutions via higher-order factorization (arXiv:1906.06196)
[1071] Chris 2020 Using constant padding, reflection padding and replication padding with keras MachineCurve (avaliable at: www.machinecurve.com/index.php/2020/02/10/Using-constant-padding-reflection-padding-and-replication-padding-with-keras)
[1072] Liu G *et al* 2018 Partial convolution based padding (arXiv:1811.11718)
[1073] Larsson G, Maire M and Shakhnarovich G FractalNet: ultra-deep neural networks without residuals (arXiv:1605.07648)
[1074] Szegedy C, Ioffe S, Vanhoucke V and Alemi A A 2017 Inception-v4, Inception-ResNet and the impact of residual connections on learning *31st Conf. on Artificial Intelligence*
[1075] Szegedy C, Vanhoucke V, Ioffe S, Shlens J and Wojna Z 2016 Rethinking the inception architecture for computer vision *Proc. Conf. on Computer Vision and Pattern Recognition* pp 2818–26
[1076] Szegedy C *et al* 2015 Going deeper with convolutions *Proc. Conf. on Computer Vision and Pattern Recognition* pp 1–9
[1077] Zoph B, Vasudevan V, Shlens J and Le Q V 2018 Learning transferable architectures for scalable image recognition *Proc. Conf. on Computer Vision and Pattern Recognition* pp 8697–8710
[1078] Kim J, Kwon Lee J and Mu Lee K 2016 Deeply-recursive convolutional network for image super-resolution *Proc. Conf. on Computer Vision and Pattern Recognition* pp 1637–45
[1079] Tai Y, Yang J and Liu X 2017 Image super-resolution via deep recursive residual network *Proc. Conf. on Computer Vision and Pattern Recognition* pp 3147–55
[1080] He K, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition *Proc. Conf. on Computer Vision and Pattern Recognition* pp 770–8
[1081] Dwarampudi M and Reddy N 2019 Effects of padding on LSTMs and CNNs (arXiv:1903.07288)
[1082] Liu G *et al* 2018 Image inpainting for irregular holes using partial convolutions *Proc. Conf. on Computer Vision (ECCV)* pp 85–100
[1083] Peng Z 2017 Multilayer perceptron algebra (arXiv:1701.04968)
[1084] Pratama M, Za'in C, Ashfahani A, Ong Y S and Ding W 2019 Automatic construction of multi-layer perceptron network from streaming examples *Proc. 28th ACM Int. Conf. on Information and Knowledge Management* pp 1171–80
[1085] Neyshabur B 2020 Towards learning convolutions from scratch (arXiv:2007.13657)
[1086] Guo L, Liu F, Cai C, Liu J and Zhang G 2019 3D deep encoder–decoder network for fluorescence molecular tomography *Opt. Lett.* **44** 1892–5
[1087] Oseledets I V 2011 Tensor-train decomposition *SIAM J. Sci. Comput.* **33** 2295–2317
[1088] Novikov A, Podoprikhin D, Osokin A and Vetrov D P 2015 Tensorizing neural networks *Advances in Neural Information Processing Systems* pp 442–50
[1089] Kong C and Lucey S 2017 Take it in your stride: do we need striding in CNNs? (arXiv:1712.02502)
[1090] Zaniolo L and Marques O 2020 On the use of variable stride in convolutional neural networks *Multimedia Tools Appl.* **79** 13581–98
[1091] Shi W *et al* 2016 Is the deconvolution layer the same as a convolutional layer? (arXiv:1609.07009)
[1092] Aitken A *et al* 2017 Checkerboard artifact free sub-pixel convolution: a note on sub-pixel convolution, resize convolution and convolution resize (arXiv:1707.02937)
[1093] Odena A, Dumoulin V and Olah C 2016 Deconvolution and checkerboard artifacts *Distill* **1**
[1094] Howard A G *et al* 2017 MobileNets: efficient convolutional neural networks for mobile vision applications (arXiv:1704.04861)
[1095] Guo J, Li Y, Lin W, Chen Y and Li J 2018 Network decoupling: from regular to depthwise separable convolutions (arXiv:1808.05517)
[1096] Depthwise Separable Convolutional Neural Networks 2020 GeeksforGeeks (available at: www.geeksforgeeks.org/depth-wise-separable-convolutional-neural-networks)
[1097] Liu T 2020 Depth-wise separable convolutions: performance investigations (available at: https://tlkh.dev/depsep-convs-perf-investigations)
[1098] Gunther L 2019 The eye *The Physics of Music and Color* (Berlin: Springer) pp 325–35
[1099] Lamb T D 2016 Why rods and cones? *Eye* **30** 179–85
[1100] Cohen A I 1972 Rods and cones *Physiology of Photoreceptor Organs* (Berlin: Springer) pp 63–110
[1101] He K, Zhang X, Ren S and Sun J 2015 Spatial pyramid pooling in deep convolutional networks for visual recognition *IEEE Trans. Pattern Anal. Mach. Intell.* **37** 1904–16
[1102] Zhang D-Q 2018 Image recognition using scale recurrent neural networks (arXiv:1803.09218)
[1103] Tanaka N 2017 Introduction to Fourier transforms for TEM and STEM *Electron Nano-Imaging* (Berlin: Springer) pp 219–26
[1104] Fourier Transform Conventions Mathematica Documentation 2020 (available at: https://reference.wolfram.com/language/tutorial/Calculus.html#26017)
[1105] Frigo M and Johnson S G 2005 The design and implementation of FFTW3 *Proc. IEEE* **93** 216–31
[1106] Stokfiszewski K, Wieloch K and Yatsymirskyy M 2017 The fast Fourier transform partitioning scheme for GPU's computation effectiveness improvement *Conf. on Computer Science and Information Technologies* (Springer) pp 511–22
[1107] Chen Y, Cui X and Mei H 2010 Large-scale FFT on GPU clusters *Proc. 24th ACM Int. Conf. on Supercomputing* pp 315–24
[1108] Gu L, Li X and Siegel J 2010 An empirically tuned 2D and 3D FFT library on CUDA GPU *Proc. 24th ACM Int. Conf. on Supercomputing* pp 305–14

[1109] Puchała D, Stokfiszewski K, Yatsymirskyy M and Szczepaniak B 2015 Effectiveness of fast Fourier transform implementations on GPU and CPU *2015 16th Int. Conf. on Computational Problems of Electrical Engineering (CPEE)* (IEEE) pp 162–4
[1110] Ogata Y, Endo T, Maruyama N and Matsuoka S 2008 An efficient, model-based CPU-GPU heterogeneous FFT library *2008 IEEE Int. Symp. on Parallel and Distributed Processing* (IEEE) pp 1–10
[1111] Cooley J W and Tukey J W 1965 An algorithm for the machine calculation of complex Fourier series *Math. Comput.* **19** 297–301
[1112] Duhamel P and Vetterli M 1990 Fast Fourier transforms: a tutorial review and a state of the art *Signal Process.* **19** 259–99
[1113] clFFT Repository 2017 (available at: https://github.com/clMathLibraries/clFFT)
[1114] Highlander T and Rodriguez A 2016 Very efficient training of convolutional neural networks using fast Fourier transform and overlap-and-add (arXiv:1601.06815)
[1115] Weisstein E W 2020 Convolution theorem Wolfram Mathworld—a Wolfram Web Resource (available at: https://mathworld.wolfram.com/ConvolutionTheorem.html)
[1116] Pratt H, Williams B, Coenen F and Zheng Y 2017 FCNN: Fourier convolutional neural networks *Joint Conf. on Machine Learning and Knowledge Discovery in Databases* (Springer) pp 786–98
[1117] Simonyan K and Zisserman A 2014 Very deep convolutional networks for large-scale image recognition (arXiv:1409.1556)
[1118] Thomson A M 2010 Neocortical layer 6, a review *Frontiers Neuroanat.* **4** 13
[1119] Fitzpatrick D 1996 The functional organization of local circuits in visual cortex: insights from the study of tree shrew striate cortex *Cereb. Cortex* **6** 329–41
[1120] Zaeemzadeh A, Rahnavard N and Shah M 2020 Norm-preservation: why residual networks can become extremely deep? *IEEE Trans. Pattern Anal. Mach. Intell.* (https://doi.org/10.1109/TPAMI.2020.2990339)
[1121] Kawaguchi K and Bengio Y 2019 Depth with nonlinearity creates no bad local minima in ResNets *Neural Netw.* **118** 167–74
[1122] Li H, Xu Z, Taylor G, Studer C and Goldstein T 2018 Visualizing the loss landscape of neural nets *Advances in Neural Information Processing Systems* pp 6389–99
[1123] Veit A, Wilber M J and Belongie S 2016 Residual networks behave like ensembles of relatively shallow networks *Advances in Neural Information Processing Systems* pp 550–8
[1124] Greff K, Srivastava R K and Schmidhuber J 2016 Highway and residual networks learn unrolled iterative estimation (arXiv:1612.07771)
[1125] Martinez J, Hossain R, Romero J and Little J J 2017 A simple yet effective baseline for 3D human pose estimation *Proc. IEEE Int. Conf. on Computer Vision* pp 2640–9
[1126] Yue B, Fu J and Liang J 2018 Residual recurrent neural networks for learning sequential representations *Information* **9** 56
[1127] Kim J, El-Khamy M and Lee J 2017 Residual LSTM: design of a deep recurrent architecture for distant speech recognition *Proc. of Interspeech 2017* pp 1591–5
[1128] Wu Y *et al* 2016 Google's neural machine translation system: bridging the gap between human and machine translation (arXiv:1609.08144)
[1129] Srivastava R K, Greff K and Schmidhuber J 2015 Training very deep networks *Advances in Neural Information Processing Systems* pp 2377–85
[1130] Srivastava R K, Greff K and Schmidhuber J 2015 Highway networks (arXiv:1505.00387)
[1131] Huang G, Liu Z, Pleiss G, Van Der Maaten L and Weinberger K 2019 Convolutional networks with dense connectivity *IEEE Trans. Pattern Anal. Mach. Intell.* (https://doi.org/10.1109/TPAMI.2019.2918284)
[1132] Huang G, Liu Z, Van Der Maaten L and Weinberger K Q 2017 Densely connected convolutional networks *Proc. Conf. on Computer Vision and Pattern Recognition* pp 4700–8
[1133] Tong T, Li G, Liu X and Gao Q 2017 Image super-resolution using dense skip connections *Proc. IEEE Int. Conf. on Computer Vision* pp 4799–807
[1134] Jiang F *et al* 2017 An end-to-end compression framework based on convolutional neural networks *IEEE Trans. Circuits Syst. Video Technol.* **28** 3007–18
[1135] Yang G and Schoenholz S 2017 Mean field residual networks: on the edge of chaos *Advances in Neural Information Processing Systems* pp 7103–14
[1136] Xiao L, Bahri Y, Sohl-Dickstein J, Schoenholz S and Pennington J 2018 Dynamical isometry and a mean field theory of CNNs: how to train 10,000-layer vanilla convolutional neural networks *Int. Conf. on Machine Learning* pp 5393–402
[1137] Wu Q and Wang F 2019 Concatenate convolutional neural networks for non-intrusive load monitoring across complex background *Energies* **12** 1572
[1138] Terwilliger A M, Perdue G N, Isele D, Patton R M and Young S R 2017 Vertex reconstruction of neutrino interactions using deep learning *2017 Int. Conf. on Neural Networks (IJCNN)* (IEEE) pp 2275–81
[1139] Gers F A, Schraudolph N N and Schmidhuber J 2002 Learning precise timing with LSTM recurrent networks *J. Mach. Learn. Res.* **3** 115–43
[1140] Gers F A and Schmidhuber E 2001 LSTM recurrent networks learn simple context-free and context-sensitive languages *IEEE Trans. Neural Netw.* **12** 1333–40
[1141] Lin M, Chen Q and Yan S 2013 Network-in-network (arXiv:1312.4400)
[1142] Vaswani A *et al* 2017 Attention is all you need *Advances in Neural Information Processing Systems* pp 5998–6008
[1143] Alammar J 2018 The illustrated transformer GitHub Blog (avaliable at: http://jalammar.github.io/illustrated-transformer)
[1144] Mnih V, Heess N, Graves A and Kavukcuoglu K 2014 Recurrent models of visual attention *Advances in Neural Information Processing Systems* pp 2204–12
[1145] Ba J, Mnih V and Kavukcuoglu K 2014 Multiple object recognition with visual attention (arXiv:1412.7755)
[1146] Lillicrap T P *et al* 2015 Continuous control with deep reinforcement learning (arXiv:1509.02971)
[1147] Heess N, Hunt J J, Lillicrap T P and Silver D 2015 Memory-based control with recurrent neural networks (arXiv:1512.04455)
[1148] Konda V R and Tsitsiklis J N 2000 Actor-critic algorithms *Advances in Neural Information Processing Systems* pp 1008–14
[1149] Grabocka J, Scholz R and Schmidt-Thieme L 2019 Learning surrogate losses (arXiv:1905.10108)
[1150] Neftci E O, Mostafa H and Zenke F 2019 Surrogate gradient learning in spiking neural networks *IEEE Signal Process. Mag.* **36** 61–3
[1151] Liang K J, Li C, Wang G and Carin L 2018 Generative adversarial network training is a continual learning problem (arXiv:1811.11083)
[1152] Jaderberg M *et al* 2017 Decoupled neural interfaces using synthetic gradients *Int. Conf. on Machine Learning* pp 1627–35
[1153] Wang Z, Bovik A C, Sheikh H R and Simoncelli E P 2004 Image quality assessment: from error visibility to structural similarity *IEEE Trans. Image Process.* **13** 600–12

[1154] Pan Z *et al* 2020 Loss functions of generative adversarial networks (GANs): opportunities and challenges *IEEE Trans. on Emerging Topics in Computational Intelligence* **4** 500–22
[1155] Dong H-W and Yang Y-H 2019 Towards a deeper understanding of adversarial losses (arXiv:1901.08753)
[1156] Mescheder L, Geiger A and Nowozin S 2018 Which training methods for gans do actually converge? (arXiv:1801.04406)
[1157] Kurach K, Lučić M, Zhai X, Michalski M and Gelly S 2019 A large-scale study on regularization and normalization in GANs *Int. Conf. on Machine Learning* pp 3581–90
[1158] Roth K, Lucchi A, Nowozin S and Hofmann T 2017 Stabilizing training of generative adversarial networks through regularization *Advances in Neural Information Processing Systems* pp 2018–28
[1159] Goodfellow I *et al* 2014 Generative adversarial nets *Advances in Neural Information Processing Systems* pp 2672–80
[1160] Mao X *et al* 2018 On the effectiveness of least squares generative adversarial networks *IEEE Trans. Pattern Anal. Mach. Intell.* **41** 2947–60
[1161] Mao X *et al* 2017 Least squares generative adversarial networks *Proc. IEEE Int. Conf. on Computer Vision* pp 2794–802
[1162] Wiatrak M and Albrecht S V 2019 Stabilizing generative adversarial network training: a survey (arXiv:1910.00927)
[1163] Bang D and Shim H 2018 MGGAN: solving mode collapse using manifold guided training (arXiv:1804.04391)
[1164] Arjovsky M, Chintala S and Bottou L 2017 Wasserstein generative adversarial networks *Int. Conf. on Machine Learning* pp 214–23
[1165] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V and Courville A C 2017 Improved training of wasserstein GANs *Advances in Neural Information Processing Systems* pp 5767–77
[1166] Hazan T, Papandreou G and Tarlow D 2017 *Adversarial Perturbations of Deep Neural Networks* (Cambridge, MA: MIT Press) pp 311–42
[1167] Chen Z, Badrinarayanan V, Lee C-Y and Rabinovich A 2017 GradNorm: gradient normalization for adaptive loss balancing in deep multitask networks (arXiv:1711.02257)
[1168] Lee S and Son Y 2020 Multitask learning with single gradient step update for task balancing (arXiv:2005.09910)
[1169] Zhang H, Goodfellow I, Metaxas D and Odena A 2019 Self-attention generative adversarial networks *Int. Conf. on Machine Learning* pp 7354–63
[1170] Brock A, Donahue J and Simonyan K 2018 Large scale GAN training for high fidelity natural image synthesis (arXiv:1809.11096)
[1171] Hindupur A 2018 The GAN zoo (available at: https://github.com/hindupuravinash/the-gan-zoo)
[1172] Wang T-C *et al* 2018 High-resolution image synthesis and semantic manipulation with conditional GANs *Proc. Conf. on Computer Vision and Pattern Recognition* pp 8798–807
[1173] Bashkirova D, Usman B and Saenko K 2018 Unsupervised video-to-video translation (arXiv:1806.03698)
[1174] Liu M-Y, Breuel T and Kautz J 2017 Unsupervised image-to-image translation networks *Advances in Neural Information Processing Systems* pp 700–8
[1175] Amodio M and Krishnaswamy S 2019 TraVeLGAN: image-to-image translation by transformation vector learning *Proc. Conf. on Computer Vision and Pattern Recognition* pp 8983–92
[1176] Tzeng E, Hoffman J, Saenko K and Darrell T 2017 Adversarial discriminative domain adaptation *Proc. Conf. on Computer Vision and Pattern Recognition* pp 7167–76
[1177] Ganin Y and Lempitsky V 2015 Unsupervised domain adaptation by backpropagation *Int. Conf. on Machine Learning* pp 1180–9
[1178] Tzeng E, Hoffman J, Darrell T and Saenko K 2015 Simultaneous deep transfer across domains and tasks *Proc. IEEE Int. Conf. on Computer Vision* pp 4068–76
[1179] Werbos P J 1990 Backpropagation through time: what it does and how to do it *Proc. IEEE* **78** 1550–60
[1180] Saldi N, Yüksel S and Linder T 2019 Asymptotic optimality of finite model approximations for partially observed markov decision processes with discounted cost *IEEE Trans. Autom. Control* **65** 130–42
[1181] Jaakkola T, Singh S P and Jordan M I 1995 Reinforcement learning algorithm for partially observable Markov decision problems *Advances in Neural Information Processing Systems* pp 345–52
[1182] Xu K *et al* 2015 Show, attend and tell: neural image caption generation with visual attention *Int. Conf. on Machine Learning* pp 2048–57
[1183] Vinyals O, Toshev A, Bengio S and Erhan D 2015 Show and tell: a neural image caption generator *Proc. Conf. on Computer Vision and Pattern Recognition* pp 3156–64
[1184] Basmatkar P, Holani H and Kaushal S 2019 Survey on neural machine translation for multilingual translation system *2019 3rd Int. Conf. on Computing Methodologies and Communication (ICCMC)* (IEEE) pp 443–8
[1185] Wu S *et al* 2020 Deep learning in clinical natural language processing: a methodical review *J. Am. Med. Inform. Assoc.* **27** 457–70
[1186] Otter D W, Medina J R and Kalita J K 2020 A survey of the usages of deep learning for natural language processing *IEEE Trans. on Neural Networks and Learning Systems* (https://doi.org/10.1109/TNNLS.2020.2979670)
[1187] Iyer S R, An U and Subramanian L 2020 Forecasting sparse traffic congestion patterns using message-passing RNNs *ICASSP 2020-2020 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 3772–6
[1188] Mandal P K and Mahto R 2019 Deep CNN-LSTM with word embeddings for news headline sarcasm detection *16th Int. Conf. on Information Technology-New Generations (ITNG 2019)* (Springer) pp 495–8
[1189] Rhanoui M, Mikram M, Yousfi S and Barzali S 2019 A CNN-BiLSTM model for document-level sentiment analysis *Machine Learning and Knowledge Extraction* **1** 832–47
[1190] Zhang X, Chen F and Huang R 2018 A combination of RNN and CNN for attention-based relation classification *Proc. Comput. Sci.* **131** 911–7
[1191] Qu Y, Liu J, Kang L, Shi Q. and Ye D 2018 Question answering over freebase via attentive RNN with similarity matrix based CNN **38** (arXiv:1804.03317)
[1192] Sieg A 2019 From pre-trained word embeddings to pre-trained language models—focus on BERT towards data science (available at: https://towardsdatascience.com/from-pre-trained-word-embeddings-to-pre-trained-language-models-focus-on-bert-343815627598)
[1193] Devlin J, Chang M-W, Lee K and Toutanova K 2018 BERT: pre-training of deep bidirectional transformers for language understanding (arXiv:1810.04805)
[1194] Mikolov T, Sutskever I, Chen K, Corrado G S and Dean J 2013 Distributed representations of words and phrases and their compositionality *Advances in Neural Information Processing Systems* pp 3111–19
[1195] Mnih A and Kavukcuoglu K 2013 Learning word embeddings efficiently with noise-contrastive estimation *Advances in Neural Information Processing Systems* pp 2265–73

[1196] Grave E, Bojanowski P, Gupta P, Joulin A and Mikolov T 2018 Learning word vectors for 157 languages *Proc. 11th Int. Conf. on Language Resources and Evaluation (LREC 2018)*
[1197] Le Q and Mikolov T 2014 Distributed representations of sentences and documents *Int. Conf. on Machine Learning* pp 1188–96
[1198] Lau J H and Baldwin T 2016 An empirical evaluation of doc2vec with practical insights into document embedding generation (arXiv:1607.05368)
[1199] Pennington J, Socher R and Manning C D 2014 GloVe: global vectors for word representation *Proc. 2014th Conf. on Empirical Methods in Natural Language Processing (EMNLP)* pp 1532–43
[1200] Mikolov T, Chen K, Corrado G and Dean J 2013 Efficient estimation of word representations in vector space (arXiv:1301.3781)
[1201] Sherstinsky A 2020 Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network *Physica* D **404** 132306
[1202] Olah C 2015 Understanding LSTM networks (available at: https://colah.github.io/posts/2015-08-Understanding-LSTMs)
[1203] Gers F A, Schmidhuber J and Cummins F 2000 Learning to forget: continual prediction with LSTM *Neural Comput.* **12** 2451–71
[1204] Hochreiter S and Schmidhuber J 1997 Long short-term memory *Neural Comput.* **9** 1735–80
[1205] Cho K *et al* 2014 Learning phrase representations using rnn encoder–decoder for statistical machine translation (arXiv:1406.1078)
[1206] Dey R and Salemt F M 2017 Gate-variants of gated recurrent unit (GRU) neural networks *2017 IEEE 60th Int. Symp. on Circuits and Systems (MWSCAS)* (IEEE) pp 1597–600
[1207] Heck J C and Salem F M 2017 Simplified minimal gated unit variations for recurrent neural networks *2017 IEEE 60th Int. Symp. on Circuits and Systems (MWSCAS)* (IEEE) pp 1593–6
[1208] Pascanu R, Mikolov T and Bengio Y 2013 On the difficulty of training recurrent neural networks *Int. Conf. on Machine Learning* pp 1310–18
[1209] Hanin B 2018 Which neural net architectures give rise to exploding and vanishing gradients? *Advances in Neural Information Processing Systems* pp 582–91
[1210] Britz D, Goldie A, Luong M-T and Le Q 2017 Massive exploration of neural machine translation architectures (arXiv:1703.03906)
[1211] Jozefowicz R, Zaremba W and Sutskever I 2015 An empirical exploration of recurrent network architectures *Int. Conf. on Machine Learning* pp 2342–50
[1212] Chung J, Gulcehre C, Cho K and Bengio Y 2014 Empirical evaluation of gated recurrent neural networks on sequence modeling *NIPS 2014 Workshop on Deep Learning*
[1213] Gruber N and Jockisch A 2020 Are GRU cells more specific and LSTM cells more sensitive in motive classification of text? *Frontiers Artif. Intell.* **3** 40
[1214] Weiss G, Goldberg Y and Yahav E 2018 On the practical computational power of finite precision RNNs for language recognition (arXiv:1805.04908)
[1215] Bayer J, Wierstra D, Togelius J and Schmidhuber J 2009 Evolving memory cell structures for sequence learning *Int. Conf. on Artificial Neural Networks* (Springer) pp 755–64
[1216] Zhou G-B, Wu J, Zhang C-L and Zhou Z-H 2016 Minimal gated unit for recurrent neural networks *Int. J. Autom. Comput.* **13** 226–34
[1217] Greff K, Srivastava R K, Koutník J, Steunebrink B R and Schmidhuber J 2016 LSTM: a search space odyssey *IEEE Trans. Neural Netw. Learn. Syst.* **28** 2222–32
[1218] Mozer M C, Kazakov D and Lindsey R V 2017 Discrete event, continuous time RNNs (arXiv:1710.04110)
[1219] Funahashi K-i and Nakamura Y 1993 Approximation of dynamical systems by continuous time recurrent neural networks *Neural Netw.* **6** 801–6
[1220] Quinn M 2001 Evolving communication without dedicated communication channels *Conf. on Artificial Life* (Springer) pp 357–66
[1221] Beer R D 1997 The dynamics of adaptive behavior: a research program *Robot. Auton. Syst.* **20** 257–89
[1222] Harvey I, Husbands P and Cliff D 1994 Seeing the light: artificial evolution, real vision *From Animals Animats* **3** 392–401
[1223] Elman J L 1990 Finding structure in time *Cogn. Sci.* **14** 179–211
[1224] Jordan M I 1997 Serial order: a parallel distributed processing approach *Advances in Psychology* vol 121 (Amsterdam: Elsevier) pp 471–95
[1225] Li S, Li W, Cook C, Zhu C and Gao Y 2018 Independently recurrent neural network (IndRNN): building a longer and deeper RNN *Proc. Conf. on Computer Vision and Pattern Recognition* pp 5457–66
[1226] Sathasivam S and Abdullah W A T W 2008 Logic learning in hopfield networks (arXiv:0804.4075)
[1227] Tutschku K 1995 Recurrent multilayer perceptrons for identification and control: the road to applications *Institute of Computer Science Research Report* (Am Hubland: University of Würzburg)
[1228] Jia Y, Wu Z, Xu Y, Ke D and Su K 2017 Long short-term memory projection recurrent neural network architectures for piano's continuous note recognition *J. Robot.* **2017**
[1229] Pascanu R, Gulcehre C, Cho K and Bengio Y 2014 How to construct deep recurrent neural networks *Proc. Second Int. Conf. on Learning Representations (ICLR 2014)*
[1230] Schuster M and Paliwal K K 1997 Bidirectional recurrent neural networks *IEEE Trans. Signal Process.* **45** 2673–81
[1231] Bahdanau D, Cho K and Bengio Y 2015 Neural machine translation by jointly learning to align and translate *3rd Int. Conf. on Learning Representations, ICLR 2015*
[1232] Graves A and Schmidhuber J 2005 Framewise phoneme classification with bidirectional lstm and other neural network architectures *Neural Netw.* **18** 602–10
[1233] Thireou T and Reczko M 2007 Bidirectional long short-term memory networks for predicting the subcellular localization of eukaryotic proteins *IEEE/ACM Trans. Comput. Biol. Bioinform.* **4** 441–6
[1234] Cho K, Van Merriënboer B, Bahdanau D and Bengio Y 2014 On the properties of neural machine translation: encoder–decoder approaches (arXiv:1409.1259)
[1235] Zhang T, Huang M and Zhao L 2018 Learning structured representation for text classification via reinforcement learning *32nd Conf. on Artificial Intelligence*
[1236] Chung J, Ahn S and Bengio Y 2016 Hierarchical multiscale recurrent neural networks (arXiv:1609.01704)
[1237] Sordoni A *et al* 2015 A hierarchical recurrent encoder–decoder for generative context-aware query suggestion *Proc. 24th ACM Int. Conf. on Information and Knowledge Management* pp 553–62
[1238] Paine R W and Tani J 2005 How hierarchical control self-organizes in artificial adaptive systems *Adapt. Behav.* **13** 211–25

[1239] Schmidhuber J 1992 Learning complex, extended sequences using the principle of history compression *Neural Comput.* **4** 234–42
[1240] Yamashita Y and Tani J 2008 Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment *PLoS Comput. Biol.* **4** e1000220
[1241] Shibata Alnajjar F, Yamashita Y and Tani J 2013 The hierarchical and functional connectivity of higher-order cognitive mechanisms: neurorobotic model to investigate the stability and flexibility of working memory *Frontiers Neurorobot.* **7** 2
[1242] Chaudhari S, Polatkan G, Ramanath R and Mithal V 2019 An attentive survey of attention models (arXiv:1904.02874)
[1243] Luong M-T, Pham H and Manning C D 2015 Effective approaches to attention-based neural machine translation (arXiv:1508.04025)
[1244] Bahdanau D, Cho K and Bengio Y 2014 Neural machine translation by jointly learning to align and translate (arXiv:1409.0473)
[1245] Graves A *et al* 2016 Hybrid computing using a neural network with dynamic external memory *Nature* **538** 471–6
[1246] Graves A, Wayne G and Danihelka I 2014 Neural turing machines (arXiv:1410.5401)
[1247] Tschannen M, Bachem O and Lucic M 2018 Recent advances in autoencoder-based representation learning (arXiv:1812.05069)
[1248] Hinton G E and Salakhutdinov R R 2006 Reducing the dimensionality of data with neural networks *Science* **313** 504–7
[1249] Kramer M A 1991 Nonlinear principal component analysis using autoassociative neural networks *AIChE J.* **37** 233–43
[1250] Zhou Y, Arpit D, Nwogu I and Govindaraju V 2014 Is joint training better for deep auto-encoders? (arXiv:1405.1380)
[1251] Jolliffe I T and Cadima J 2016 Principal component analysis: a review and recent developments *Phil. Trans. R. Soc.* A **374** 20150202
[1252] Theis L, Shi W, Cunningham A and Huszár F 2017 Lossy image compression with compressive autoencoders (arXiv:1703.00395)
[1253] Vincent P *et al* 2010 Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion *J. Mach. Learn. Res.* **11** 3371–408
[1254] Vincent P, Larochelle H, Bengio Y and Manzagol P-A 2008 Extracting and composing robust features with denoising autoencoders *Proc. 25th Int. Conf. on Machine Learning* pp 1096–103
[1255] Gondara L 2016 Medical image denoising using convolutional denoising autoencoders *2016 IEEE 16th Int. Conf. on Data Mining Workshops (ICDMW)* (IEEE) pp 241–6
[1256] Cho K 2013 Simple sparsification improves sparse denoising autoencoders in denoising highly corrupted images *Int. Conf. on Machine Learning* pp 432–40
[1257] Cho K 2013 Boltzmann machines and denoising autoencoders for image denoising (arXiv:1301.3468)
[1258] Rifai S, Vincent P, Muller X, Glorot X and Bengio Y 2011 Contractive auto-encoders: explicit invariance during feature extraction *Int. Conf. on Machine Learning*
[1259] Rifai S *et al* 2011 Higher order contractive auto-encoder *Joint European Conf. on Machine Learning and Knowledge Discovery in Databases* (Springer) pp 645–60
[1260] Kingma D P and Welling M 2019 An introduction to variational autoencoders (arXiv:1906.02691)
[1261] Doersch C 2016 Tutorial on variational autoencoders (arXiv:1606.05908)
[1262] Makhzani A and Frey B 2013 *k*-sparse autoencoders (arXiv:1312.5663)
[1263] Nair V and Hinton G E 2009 3D object recognition with deep belief nets *Advances in Neural Information Processing Systems* pp 1339–47
[1264] Arpit D, Zhou Y, Ngo H and Govindaraju V 2016 Why regularized auto-encoders learn sparse representation? *Int. Conf. on Machine Learning* pp 136–44
[1265] Zeng N *et al* 2018 Facial expression recognition via learning deep sparse autoencoders *Neurocomputing* **273** 643–9
[1266] Yin Y, Ouyang L, Wu Z and Yin S 2020 A survey of generative adversarial networks based on encoder–decoder model *Math. Comput. Sci.* **5** 31
[1267] Yu X, Zhang X, Cao Y and Xia M 2019 VAEGAN: a collaborative filtering framework based on adversarial variational autoencoders *Proc. 28th Int. Conf. on Artificial Intelligence* (AAAI Press) pp 4206–12
[1268] Larsen A B L, Sønderby S K, Larochelle H and Winther O 2016 Autoencoding beyond pixels using a learned similarity metric *Int. Conf. on Machine Learning* pp 1558–66
[1269] Zhuang F and Moulin P 2020 A new variational method for deep supervised semantic image hashing *ICASSP 2020–2020 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 4532–6
[1270] Jin G, Zhang Y and Lu K 2019 Deep hashing based on VAE-GAN for efficient similarity retrieval *Chin. J. Electron.* **28** 1191–7
[1271] Khobahi S and Soltanalian M 2019 Model-aware deep architectures for one-bit compressive variational autoencoding (arXiv:1911.12410)
[1272] Wang B, Liu K and Zhao J 2018 Deep semantic hashing with multi-adversarial training *Proc. 27th ACM Int. Conf. on Information and Knowledge Management* pp 1453–62
[1273] Patterson N and Wang Y 2016 Semantic hashing with variational autoencoders
[1274] Fan Y *et al* 2020 Video anomaly detection and localization via gaussian mixture fully convolutional variational autoencoder *Comput. Vis. Image Underst.* **195** 102920
[1275] Yao R, Liu C, Zhang L and Peng P 2019 Unsupervised anomaly detection using variational auto-encoder based feature extraction *2019 IEEE Int. Conf. on Prognostics and Health Management (ICPHM)* (IEEE) pp 1–7
[1276] Xu H *et al* 2018 Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications *Proc. 2018 World Wide Conf.* pp 187–96
[1277] An J and Cho S 2015 Variational autoencoder based anomaly detection using reconstruction probability *Spec. Lecture IE* **2** 1–18
[1278] Gauerhof L and Gu N 2020 Reverse variational autoencoder for visual attribute manipulation and anomaly detection *2020 IEEE Conf. on Applications of Computer Vision (WACV)* (IEEE) pp 2103–12
[1279] Klys J, Snell J and Zemel R 2018 Learning latent subspaces in variational autoencoders *Advances in Neural Information Processing Systems* pp 6444–54
[1280] Borysov S S, Rich J and Pereira F C 2019 How to generate micro-agents? A deep generative modeling approach to population synthesis *Transp. Res.* C **106** 73–97
[1281] Salim Jr A 2018 Synthetic patient generation: a deep learning approach using variational autoencoders (arXiv:1808.06444)
[1282] Gómez-Bombarelli R *et al* 2018 Automatic chemical design using a data-driven continuous representation of molecules *ACS Cent. Sci.* **4** 268–76
[1283] Zhavoronkov A *et al* 2019 Deep learning enables rapid identification of potent DDR1 kinase inhibitors *Nat. Biotechnol.* **37** 1038–40
[1284] Griffiths R-R and Hernández-Lobato J M 2020 Constrained bayesian optimization for automatic chemical design using variational autoencoders *Chemical Science* **11** 577–86

[1285] Lim J, Ryu S, Kim J W and Kim W Y 2018 Molecular generative model based on conditional variational autoencoder for de novo molecular design *J. Cheminform.* **10** 1–9
[1286] Wan Z, Zhang Y and He H 2017 Variational autoencoder based synthetic data generation for imbalanced learning *2017 Symp. Series on Computational Intelligence (SSCI)* (IEEE) pp 1–7
[1287] Zhang J M, Harman M, Ma L and Liu Y 2020 Machine learning testing: survey, landscapes and horizons *IEEE Trans. Softw. Eng.*
[1288] Amershi S *et al* 2019 Software engineering for machine learning: a case study *2019 IEEE/ACM 41st Int. Conf. on Software Engineering: Software Engineering in Practice (ICSE-SEIP)* (IEEE) pp 291–300
[1289] Breck E, Cai S, Nielsen E, Salib M and Sculley D 2017 The ML test score: a rubric for ML production readiness and technical debt reduction *2017 IEEE Int. Conf. on Big Data (Big Data)* (IEEE) pp 1123–32
[1290] Sculley D *et al* 2015 Hidden technical debt in machine learning systems *Advances in Neural Information Processing Systems* pp 2503–11
[1291] Li H, Xu Z, Taylor G, Studer C and Goldstein T 2017 Loss landscape mit license (available at: https://github.com/tomgoldstein/loss-landscape/blob/master/LICENSE)
[1292] Rodríguez O H and Lopez Fernandez J M 2010 A semiotic reflection on the didactics of the chain rule *Math. Enthusiast* **7** 321–32
[1293] Kiefer J and Wolfowitz J 1952 Stochastic estimation of the maximum of a regression function *Ann. Math. Stat.* **23** 462–6
[1294] Robbins H and Monro S 1951 A stochastic approximation method *Ann. Math. Stat.* **22** 400–7
[1295] Polyak B T 1964 Some methods of speeding up the convergence of iteration methods *USSR Comput. Math. Math. Phys.* **4** 1–17
[1296] Sutskever I, Martens J, Dahl G and Hinton G 2013 On the importance of initialization and momentum in deep learning *Int. Conf. on Machine Learning* pp 1139–47
[1297] Su W, Boyd S and Candes E 2014 A differential equation for modeling Nesterov's accelerated gradient method: theory and insights *Advances in Neural Information Processing Systems* pp 2510–18
[1298] TensorFlow Source Code for Nesterov Momentum 2018 (available at: https://github.com/tensorflow/tensorflow/blob/23c218785eac5bfe737eec4f8081fd0ef8e0684d/tensorflow/python/training/momentum_test.py#L40)
[1299] Ma J and Yarats D 2018 Quasi-qyperbolic momentum and ADAM for deep learning (arXiv:1810.06801)
[1300] Lucas J, Sun S, Zemel R and Grosse R 2018 Aggregated momentum: stability through passive damping (arXiv:1804.00325)
[1301] Hinton G, Srivastava N and Swersky K 2012 Neural networks for machine learning lecture 6a overview of mini-batch gradient descent (available at: www.cs.toronto.edu/ tijmen/csc321/slides/lecture_slides_lec6.pdf)
[1302] Kingma D P and Ba J 2014 ADAM: a method for stochastic optimization (arXiv:1412.6980)
[1303] Sun S, Cao Z, Zhu H and Zhao J 2019 A survey of optimization methods from a machine learning perspective *IEEE Trans. Cybern.* **50** 3668–81
[1304] Bottou L, Curtis F E and Nocedal J 2018 Optimization methods for large-scale machine learning *SIAM Rev.* **60** 223–311
[1305] Ruder S 2016 An overview of gradient descent optimization algorithms (arXiv:1609.04747)
[1306] Curry H B 1944 The method of steepest descent for non-linear minimization problems *Q. Appl. Math.* **2** 258–61
[1307] Lemaréchal C 2012 Cauchy and the gradient method *Doc. Math. Extra* **251** 254
[1308] Chen T, Xu B, Zhang C and Guestrin C 2016 Training deep nets with sublinear memory cost (arXiv:1604.06174)
[1309] Cybertron AI 2019 Saving memory using gradient-checkpointing (available at: https://github.com/cybertronai/gradient-checkpointing)
[1310] Jin P, Ginsburg B and Keutzer K 2018 Spatially parallel convolutions OpenReview.net
[1311] Whittington J C R and Bogacz R 2019 Theories of error back-propagation in the brain *Trends Cogn. Sci.* **23** 235–50
[1312] Green C S and Bavelier D 2008 Exercising your brain: a review of human brain plasticity and training-induced learning *Psychol. Aging* **23** 692
[1313] Bassett D S *et al* 2011 Dynamic reconfiguration of human brain networks during learning *Proc. Natl Acad. Sci.* **108** 7641–6
[1314] O'Doherty J P 2004 Reward representations and reward-related learning in the human brain: insights from neuroimaging *Curr. Opin. Neurobiol.* **14** 769–76
[1315] Luo L, Xiong Y, Liu Y and Sun X 2019 Adaptive gradient methods with dynamic bound of learning rate (arXiv:1902.09843)
[1316] Reddi S J, Kale S and Kumar S 2019 On the convergence of ADAM and beyond (arXiv:1904.09237)
[1317] Zhang M, Lucas J, Ba J and Hinton G E 2019 Lookahead optimizer: *k* steps forward, 1 step back *Advances in Neural Information Processing Systems* pp 9597–608
[1318] Dozat T 2016 Incorporating Nesterov momentum into ADAM OpenReview (available at: https://openreview.net/forum?id=OM0jvwB8jIp57ZJjtNEZ)
[1319] Huang H, Wang C and Dong B 2018 Nostalgic Adam: weighting more of the past gradients when designing the adaptive learning rate (arXiv:1805.07557)
[1320] Baiesi M 2019 Power gradient descent (arXiv:1906.04787)
[1321] Liu L *et al* 2019 On the variance of the adaptive learning rate and beyond (arXiv:1908.03265)
[1322] Bello I, Zoph B, Vasudevan V and Le Q V 2017 Neural optimizer search with reinforcement learning (arXiv:1709.07417)
[1323] Andrychowicz M *et al* 2016 Learning to learn by gradient descent by gradient descent *Advances in Neural Information Processing Systems* pp 3981–9
[1324] Li K and Malik J 2016 Learning to optimize (arXiv:1606.01885)
[1325] Hochreiter S, Younger A S and Conwell P R 2001 Learning to learn using gradient descent *Int. Conf. on Artificial Neural Networks* (Springer) pp 87–94
[1326] Duan Y *et al* 2016 RL^2: fast reinforcement learning via slow reinforcement learning (arXiv:1611.02779)
[1327] Zou D, Cao Y, Zhou D and Gu Q 2018 Stochastic gradient descent optimizes over-parameterized deep ReLU networks (arXiv:1811.08888)
[1328] Watt J 2020 Two natural weaknesses of gradient descent (available at: https://jermwatt.github.io/machine_learning_refined/notes/3_First_order_methods/3_7_Problems.html)
[1329] Goh G 2017 Why momentum really works *Distill* (https://doi.org/10.23915/distill.00006)
[1330] Qian N 1999 On the momentum term in gradient descent learning algorithms *Neural Netw.* **12** 145–51
[1331] Schmidt R M, Schneider F and Hennig P 2020 Descending through a crowded valley—benchmarking deep learning optimizers (arXiv:2007.01547)
[1332] Choi D *et al* 2019 On empirical comparisons of optimizers for deep learning (arXiv:1910.05446)
[1333] Wilson A C, Roelofs R, Stern M, Srebro N and Recht B 2017 The marginal value of adaptive gradient methods in machine learning *Advances in Neural Information Processing Systems* pp 4148–58

[1334] Dogo E, Afolabi O, Nwulu N, Twala B and Aigbavboa C 2018 A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks *2018 Int. Conf. on Computational Techniques, Electronics and Mechanical Systems (CTEMS)* (IEEE) pp 92–9
[1335] Seetharaman P, Wichern G, Pardo B and Roux J L 2020 AutoClip: adaptive gradient clipping for source separation networks (arXiv:2007.14469)
[1336] Gorbunov E, Danilova M and Gasnikov A 2020 Stochastic optimization with heavy-tailed noise via accelerated gradient clipping (arXiv:2005.10785)
[1337] Yoshida T and Ohki K 2020 Natural images are reliably represented by sparse and variable populations of neurons in visual cortex *Nat. Commun.* **11** 1–19
[1338] Probst P, Bischl B and Boulesteix A-L 2018 Tunability: importance of hyperparameters of machine learning algorithms (arXiv:1802.09596)
[1339] Ge R, Kakade S M, Kidambi R and Netrapalli P 2019 The step decay schedule: a near optimal, geometrically decaying learning rate procedure (arXiv:1904.12838)
[1340] Chen J and Kyrillidis A 2019 Decaying momentum helps neural network training (arXiv:1910.04952)
[1341] Yang L and Shami A 2020 On hyperparameter optimization of machine learning algorithms: theory and practice (arXiv:2007.15745)
[1342] Chandra K *et al* 2019 Gradient descent: the ultimate optimizer (arXiv:1909.13371)
[1343] Akiba T, Sano S, Yanase T, Ohta T and Koyama M 2019 Optuna: a next-generation hyperparameter optimization framework *Proc. 25th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* pp 2623–31
[1344] Lakhmiri D, Digabel S L and Tribes C 2019 HyperNOMAD: hyperparameter optimization of deep neural networks using mesh adaptive direct search (arXiv:1907.01698)
[1345] Ilievski I, Akhtar T, Feng J and Shoemaker C A 2017 Efficient hyperparameter optimization of deep learning algorithms using deterministic RBF surrogates *Proc. 31st Conf. on Artificial Intelligence* (AAAI Press) pp 822–9
[1346] Lorenzo P R, Nalepa J, Kawulok M, Ramos L S and Pastor J R 2017 Particle swarm optimization for hyper-parameter selection in deep neural networks *Proc. Genetic and Evolutionary Conf.* pp 481–8
[1347] Wilamowski B M and Yu H 2010 Neural network learning without backpropagation *IEEE Trans. Neural Netw.* **21** 1793–803
[1348] Blum A, Dan C and Seddighin S 2020 Learning complexity of simulated annealing (arXiv:2003.02981)
[1349] Ingber L 1993 Simulated annealing: practice versus theory *Math. Comput. Modelling* **18** 29–57
[1350] Ayumi V, Rere L R, Fanany M I and Arymurthy A M 2016 Optimization of convolutional neural network using microcanonical annealing algorithm *2016 Int. Conf. on Advanced Computer Science and Information Systems (ICACSIS)* (IEEE) pp 506–11
[1351] Rere L M R, Fanany M I and Arymurthy A M 2015 Simulated annealing algorithm for deep learning *Proc. Comput. Sci.* **72** 137–44
[1352] Borysenko O and Byshkin M 2020 CoolMomentum: a method for stochastic optimization by langevin dynamics with simulated annealing (arXiv:2005.14605)
[1353] Fischetti M and Stringher M 2019 Embedded hyper-parameter tuning by simulated annealing (arXiv:1906.01504)
[1354] Sloss A N and Gustafson S 2020 2019 evolutionary algorithms review *Genetic Programming Theory and Practice XVII* (Berlin: Springer) pp 307–44
[1355] Al-Sahaf H *et al* 2019 A survey on evolutionary machine learning *J. R. Soc. New Zealand* **49** 205–28
[1356] Shapiro J 1999 Genetic algorithms in machine learning *Advanced Course on Artificial Intelligence* (Berlin: Springer) pp 146–68
[1357] Doerr B, Le H P, Makhmara R and Nguyen T D 2017 Fast genetic algorithms *Proc. Genetic and Evolutionary Conf* pp 777–84
[1358] Such F P *et al* 2017 Deep neuroevolution: genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning (arXiv:1712.06567)
[1359] Sehgal A, La H, Louis S and Nguyen H 2019 Deep reinforcement learning using genetic algorithm for parameter optimization *2019 3rd IEEE Int. Conf. on Robotic Computing (IRC)* (IEEE) pp 596–601
[1360] Hu C, Zuo Y, Chen C, Ong S P and Luo J 2020 Genetic algorithm-guided deep learning of grain boundary diagrams: addressing the challenge of five degrees of freedom *Mater. Today* **38** 49–57
[1361] Jennings P C, Lysgaard S, Hummelshøj J S, Vegge T and Bligaard T 2019 Genetic algorithms for computational materials discovery accelerated by machine learning *npj Comput. Mater.* **5** 1–6
[1362] Nigam A, Friederich P, Krenn M and Aspuru-Guzik A 2019 Augmenting genetic algorithms with deep neural networks for exploring the chemical space (arXiv:1909.11655)
[1363] Potapov A and Rodionov S 2017 Genetic algorithms with DNN-based trainable crossover as an example of partial specialization of general search *Int. Conf. on Artificial General Intelligence* (Springer) pp 101–11
[1364] Powell M J 1998 Direct search algorithms for optimization calculations *Numer.* **7** 287–336
[1365] Ranganathan V and Natarajan S 2018 A new backpropagation algorithm without gradient descent (arXiv:1802.00027)
[1366] Junior F E F and Yen G G 2019 Particle swarm optimization of deep neural networks architectures for image classification *Swarm Evolutionary Comput.* **49** 62–74
[1367] Qolomany B, Maabreh M, Al-Fuqaha A, Gupta A and Benhaddou D 2017 Parameters optimization of deep learning models using particle swarm optimization *2017 13th Int. Wireless Communications and Mobile Conf. (IWCMC)* (IEEE) pp 1285–90
[1368] Kennedy J and Eberhart R 1995 Particle swarm optimization *Proc. of ICNN'95—Int. Conf. on Neural Networks* vol 4 (IEEE) pp 1942–8
[1369] Kennedy J 1997 The particle swarm: social adaptation of knowledge *Proc. 1997 IEEE Int. Conf. on Evolutionary Computation (ICEC'97)* (IEEE) pp 303–8
[1370] Xu Y 2020 A review of machine learning with echo state networks *Project Report*
[1371] Jaeger H 2007 Echo state network *Scholarpedia* **2** 2330
[1372] Gallicchio C and Micheli A 2017 Deep echo state network (DeepESN): a brief survey (arXiv:1712.04323)
[1373] Alaba P A *et al* 2019 Towards a more efficient and cost-sensitive extreme learning machine: a state-of-the-art review of recent trend *Neurocomputing* **350** 70–90
[1374] Ghosh S *et al* 2018 A survey on extreme learning machine and evolution of its variants *Int. Conf. on Recent Trends in Image Processing and Pattern Recognition* (Springer) pp 572–83
[1375] Albadra M A A and Tiuna S 2017 Extreme learning machine: a review *Int. J. Appl. Eng. Res.* **12** 4610–23
[1376] Tang J, Deng C and Huang G-B 2015 Extreme learning machine for multilayer perceptron *IEEE Trans. Neural Netw. Learn. Syst.* **27** 809–21

[1377] Huang G-B, Zhou H, Ding X and Zhang R 2011 Extreme learning machine for regression and multiclass classification *IEEE Trans. Syst. Man Cybern.* B **42** 513–29
[1378] Huang G-B, Zhu Q-Y and Siew C-K 2006 Extreme learning machine: theory and applications *Neurocomputing* **70** 489–501
[1379] Huang G-B, Zhu Q-Y and Siew C-K 2004 Extreme learning machine: a new learning scheme of feedforward neural networks *2004 IEEE Int. Conf. on Neural Networks (IEEE Cat. No. 04CH37541)* vol 2 (IEEE) pp 985–90
[1380] Li Y 2017 Deep reinforcement learning: an overview (arXiv:1701.07274)
[1381] Mondal A K and Jamali N 2020 A survey of reinforcement learning techniques: strategies, recent development, and future directions (arXiv:2001.06921)
[1382] Haney B S 2020 Deep reinforcement learning patents: an empirical survey Available at SSRN 3570254
[1383] Nguyen T T, Nguyen N D and Nahavandi S 2020 Deep reinforcement learning for multiagent systems: a review of challenges, solutions and applications *IEEE Trans. Cybern.* **50** 3826–39
[1384] Botvinick M *et al* 2019 Reinforcement learning, fast and slow *Trends Cogn. Sci.* **23** 408–22
[1385] Recht B 2019 A tour of reinforcement learning: the view from continuous control *Annu. Rev. Control Robot. Auton. Syst.* **2** 253–79
[1386] Arulkumaran K, Deisenroth M P, Brundage M and Bharath A A 2017 A brief survey of deep reinforcement learning (arXiv:1708.05866)
[1387] Kiran B R *et al* 2020 Deep reinforcement learning for autonomous driving: a survey (arXiv:2002.00444)
[1388] Nageshrao S, Tseng H E and Filev D 2019 Autonomous highway driving using deep reinforcement learning *2019 IEEE Int. Conf. on Systems, Man and Cybernetics (SMC)* (IEEE) pp 2326–31
[1389] Talpaert V *et al* 2019 Exploring applications of deep reinforcement learning for real-world autonomous driving systems (arXiv:1901.01536)
[1390] Luong N C *et al* 2019 Applications of deep reinforcement learning in communications and networking: a survey *IEEE Commun. Surv. Tutorials* **21** 3133–74
[1391] Di Felice M, Bedogni L and Bononi L 2018 *Reinforcement Learning-Based Spectrum Management for Cognitive Radio Networks: A Literature Review and Case Study* (Singapore: Springer Singapore) pp 1–38
[1392] Han M *et al* 2019 A review of reinforcement learning methodologies for controlling occupant comfort in buildings *Sustain. Cities Soc.* **51** 101748
[1393] Mason K and Grijalva S 2019 A review of reinforcement learning for autonomous building energy management *Comput. Electr. Eng.* **78** 300–12
[1394] Mnih V *et al* 2015 Human-level control through deep reinforcement learning *Nature* **518** 529–33
[1395] Nguyen H and La H 2019 Review of deep reinforcement learning for robot manipulation *2019 3rd IEEE Int. Conf. on Robotic Computing (IRC)* (IEEE) pp 590–5
[1396] Bhagat S, Banerjee H, Ho Tse Z T and Ren H 2019 Deep reinforcement learning for soft, flexible robots: brief review with impending challenges *Robotics* **8** 4
[1397] Zhao T, Hachiya H, Niu G and Sugiyama M 2011 Analysis and improvement of policy gradient estimation *Advances in Neural Information Processing Systems* pp 262–70
[1398] Weng L 2020 Exploration strategies in deep reinforcement learning (available at: https://lilianweng.github.io/lil-log/2020/06/07/exploration-strategies-in-deep-reinforcement-learning.html)
[1399] Plappert M *et al* 2018 Parameter space noise for exploration (arXiv:1706.01905)
[1400] Uhlenbeck G E and Ornstein L S 1930 On the theory of the brownian motion *Phys. Rev.* **36** 823
[1401] Fujimoto S, Van Hoof H and Meger D 2018 Addressing function approximation error in actor-critic methods (arXiv:1802.09477)
[1402] Barth-Maron G *et al* 2018 Distributed distributional deterministic policy gradients (arXiv:1804.08617)
[1403] Kosaka N 2019 Has it explored enough? Master's Thesis Royal Holloway University of London
[1404] Fortunato M *et al* 2019 Noisy networks for exploration (arXiv:1706.10295)
[1405] Hazan E, Kakade S, Singh K and Van Soest A 2019 Provably efficient maximum entropy exploration *Int. Conf. on Machine Learning* pp 2681–91
[1406] Haarnoja T, Tang H, Abbeel P and Levine S 2017 Reinforcement learning with deep energy-based policies *Proc. 34th Int. Conf. on Machine Learning* vol 70 pp 1352–61
[1407] Ahmed Z, Le Roux N, Norouzi M and Schuurmans D 2019 Understanding the impact of entropy on policy optimization *Int. Conf. on Machine Learning* pp 151–60
[1408] Aubret A, Matignon L and Hassas S 2019 A survey on intrinsic motivation in reinforcement learning (arXiv:1908.06976)
[1409] Linke C, Ady N M, White M, Degris T and White A 2019 Adapting behaviour via intrinsic reward: a survey and empirical study (arXiv:1906.07865)
[1410] Pathak D, Agrawal P, Efros A A and Darrell T 2017 Curiosity-driven exploration by self- supervised prediction *Proc. Conf. on Computer Vision and Pattern Recognition Workshops* pp 16–17
[1411] Hoi S C, Sahoo D, Lu J and Zhao P 2018 Online learning: a comprehensive survey (arXiv:1802.02871)
[1412] Wei C-Y, Hong Y-T and Lu C-J 2017 Online reinforcement learning in stochastic games *Advances in Neural Information Processing Systems* pp 4987–97
[1413] Levine S, Kumar A, Tucker G and Fu J 2020 Offline reinforcement learning: tutorial, review, and perspectives on open problems (arXiv:2005.01643)
[1414] Seita D 2020 Offline (batch) reinforcement learning: a review of literature and applications Seita's place (available at: https://danieltakeshi.github.io/2020/06/28/offline-rl)
[1415] Fedus W *et al* 2020 Revisiting fundamentals of experience replay (arXiv:2007.06700)
[1416] Nair A, Dalal M, Gupta A and Levine S 2020 Accelerating online reinforcement learning with offline datasets (arXiv:2006.09359)
[1417] Lin L-J 1992 Self-improving reactive agents based on reinforcement learning, planning and teaching *Mach. Learn.* **8** 293–321
[1418] Zhang S and Sutton R S 2017 A deeper look at experience replay (arXiv:1712.01275)
[1419] He X, Zhao K and Chu X 2019 AutoML: a survey of the state-of-the-art (arXiv:1908.00709)
[1420] Malekhosseini E, Hajabdollahi M, Karimi N and Samavi S 2019 Modeling neural architecture search methods for deep networks (arXiv:1912.13183)
[1421] Jaafra Y, Laurent J L, Deruyver A and Naceur M S 2019 Reinforcement learning for neural architecture search: a review *Image Vis. Comput.* **89** 57–66
[1422] Elsken T, Metzen J H and Hutter F 2018 Neural architecture search: a survey (arXiv:1808.05377)

[1423] Waring J, Lindvall C and Umeton R 2020 Automated machine learning: review of the state-of-the-art and opportunities for healthcare *Artif. Intell. Med.* **104** 101822
[1424] Weill C *et al* 2019 AdaNet: a scalable and flexible framework for automatically learning ensembles (arXiv:1905.00080)
[1425] Weill C 2018 Introducing AdaNet: fast and flexible AutoML with learning guarantees google AI blog (available at: https://ai.googleblog.com/2018/10/introducing-adanet-fast-and-flexible.html)
[1426] Liu C *et al* 2019 Auto-DeepLab: hierarchical neural architecture search for semantic image segmentation *Proc. Conf. on Computer Vision and Pattern Recognition* pp 82–92
[1427] Gong X, Chang S, Jiang Y and Wang Z 2019 AutoGAN: neural architecture search for generative adversarial networks *Proc. IEEE Int. Conf. on Computer Vision* pp 3224–34
[1428] Jin H, Song Q and Hu X 2019 Auto-Keras: an efficient neural architecture search system *Proc. 25th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* pp 1946–56
[1429] Feurer M *et al* 2015 Efficient and robust automated machine learning *Advances in Neural Information Processing Systems* pp 2962–70
[1430] Liang H *et al* 2019 DARTS+: improved differentiable architecture search with early stopping (arXiv:1909.06035)
[1431] LeDell E and Poirier S 2020 H2O AutoML: scalable automatic machine learning *Proc. AutoML Workshop at ICML* vol 2020
[1432] Molino P, Dudin Y and Miryala S S 2019 Ludwig: a type-based declarative deep learning toolbox (arXiv:1909.07930)
[1433] Young S R, Rose D C, Karnowski T P, Lim S-H and Patton R M 2015 Optimizing deep learning hyper-parameters through an evolutionary algorithm *Proc. Workshop on Machine Learning in High-Performance Computing Environments* pp 1–5
[1434] Patton R M *et al* 2018 167-PFLOPS deep learning for electron microscopy: from learning physics to atomic manipulation *SC18: Int. Conf. for High Performance Computing, Networking, Storage and Analysis* (IEEE) pp 638–48
[1435] Kandasamy K, Neiswanger W, Schneider J, Poczos B and Xing E P 2018 Neural architecture search with bayesian optimisation and optimal transport *Advances in Neural Information Processing Systems* pp 2016–25
[1436] Nayman N *et al* 2019 XNAS: neural architecture search with expert advice *Advances in Neural Information Processing Systems* pp 1977–87
[1437] Jiang W *et al* 2019 Accuracy vs. efficiency: achieving both through FPGA-implementation aware neural architecture search *Proc. 56th Annual Design Conf. 2019* pp 1–6
[1438] Liu C *et al* 2018 Progressive neural architecture search *Proc. Conf. on Computer Vision (ECCV)* pp 19–34
[1439] Zhang C, Ren M and Urtasun R 2018 Graph hypernetworks for neural architecture search (arXiv:1810.05749)
[1440] Baker B, Gupta O, Raskar R and Naik N 2017 Accelerating neural architecture search using performance prediction (arXiv:1705.1082)
[1441] Zoph B and Le Q V 2016 Neural architecture search with reinforcement learning (arXiv:1611.01578)
[1442] Hanussek M, Blohm M and Kintz M 2020 Can AutoML outperform humans? An evaluation on popular OpenML datasets using AutoML benchmark (arXiv:2009.01564)
[1443] Godoy D 2018 Hyper-parameters in action! Part II—weight initializers towards data science (available at: https://towardsdatascience.com/hyper-parameters-in-action-part-ii-weight-initializers-35aee1a28404)
[1444] Nagarajan V and Kolter J Z 2019 Generalization in deep networks: the role of distance from initialization (arXiv:1901.01672)
[1445] Glorot X and Bengio Y 2010 Understanding the difficulty of training deep feedforward neural networks *Proc. 13th Int. Conf. on Artificial Intelligence and Statistics* pp 249–56
[1446] Kumar S K 2017 On weight initialization in deep neural networks (arXiv:1704.08863)
[1447] Saxe A M, McClelland J L and Ganguli S 2013 Exact solutions to the nonlinear dynamics of learning in deep linear neural networks (arXiv:1312.6120)
[1448] Henaff M, Szlam A and LeCun Y 2016 Recurrent orthogonal networks and long-memory tasks (arXiv:1602.06662)
[1449] Le Q V, Jaitly N and Hinton G E 2015 A simple way to initialize recurrent networks of rectified linear units (arXiv:1504.00941)
[1450] Mikolov T, Joulin A, Chopra S, Mathieu M and Ranzato M 2014 Learning longer memory in recurrent neural networks (arXiv:1412.7753)
[1451] Pitis S 2016 Non-zero initial states for recurrent neural networks (available at: https://r2rt.com/non-zero-initial-states-for-recurrent-neural-networks.html)
[1452] Mishkin D and Matas J 2015 All you need is a good init (arXiv:1511.06422)
[1453] Sussillo D and Abbott L 2014 Random walk initialization for training very deep feedforward networks (arXiv:1412.6558)
[1454] Dauphin Y N and Schoenholz S 2019 MetaInit: initializing learning by learning to initialize *Advances in Neural Information Processing Systems* pp 12645–57
[1455] Kukačka J, Golkov V and Cremers D 2017 Regularization for deep learning: a taxonomy (arXiv:1710.10686)
[1456] Kang G 2019 Regularization in deep neural networks PhD Thesis University of Technology Sydney
[1457] Liu Z, Li X, Kang B and Darrell T 2019 Regularization matters in policy optimization (arXiv:1910.09191)
[1458] Vettam S and John M 2019 Regularized deep learning with non-convex penalties (arXiv:1909.05142)
[1459] Golatkar A S, Achille A and Soatto S 2019 Time matters in regularizing deep networks: weight decay and data augmentation affect early learning dynamics, matter little near convergence *Advances in Neural Information Processing Systems* pp 10678–88
[1460] Tanay T and Griffin L D 2018 A new angle on L2 regularization (arXiv:1806.11186)
[1461] Van Laarhoven T 2017 L2 regularization versus batch and weight normalization (arXiv:1706.05350)
[1462] Van Den Doel K, Ascher U and Haber E 2012 The lost honour of L2-based regularization *Large Scale Inverse Problems* **13** 181–203
[1463] Gribonval R, Cevher V and Davies M E 2012 Compressible distributions for high-dimensional statistics *IEEE Trans. Inf. Theory* **58** 5016–34
[1464] Ng A Y 2004 Feature selection, L1 vs. L2 regularization and rotational invariance *Proc. 21st Int. Conf. on Machine Learning* p 78
[1465] Zou H and Hastie T 2005 Regularization and variable selection via the elastic net *J. R. Stat. Soc.* B **67** 301–20
[1466] Tibshirani R 1996 Regression shrinkage and selection via the lasso *J. R. Stat. Soc.* B **58** 267–88
[1467] Hoerl A E and Kennard R W 1970 Ridge regression: biased estimation for nonorthogonal problems *Technometrics* **12** 55–67
[1468] Zhang J, He T, Sra S and Jadbabaie A 2019 Why gradient clipping accelerates training: a theoretical justification for adaptivity (arXiv:1905.11881)
[1469] Chen X, Wu Z S and Hong M 2020 Understanding gradient clipping in private SGD: a geometric perspective (arXiv:2006.15429)
[1470] Menon A K, Rawat A S, Reddi S J and Kumar S 2019 Can gradient clipping mitigate label noise? *Int. Conf. on Learning Representations*

[1471] Bengio Y, Boulanger-Lewandowski N and Pascanu R 2013 Advances in optimizing recurrent networks *2013 IEEE Int. Conf. on Acoustics, Speech and Signal Processing* (IEEE) pp 8624–8
[1472] Chen M X *et al* 2018 The best of both worlds: combining recent advances in neural machine translation (arXiv:1804.09849)
[1473] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: a simple way to prevent neural networks from overfitting *J. Mach. Learn. Res.* **15** 1929–58
[1474] Labach A, Salehinejad H and Valaee S 2019 Survey of dropout methods for deep neural networks (arXiv:1904.13310)
[1475] Li Z, Gong B and Yang T 2016 Improved dropout for shallow and deep learning *Advances in Neural Information Processing Systems* pp 2523–31
[1476] Mianjy P, Arora R and Vidal R 2018 On the implicit bias of dropout *Int. Conf. on Machine Learning* pp 3540–8
[1477] Warde-Farley D, Goodfellow I J, Courville A and Bengio Y 2013 An empirical analysis of dropout in piecewise linear networks (arXiv:1312.6197)
[1478] Garbin C, Zhu X and Marques O 2020 Dropout vs. batch normalization: an empirical study of their impact to deep learning *Multimedia Tools Appl.* **79** 12777–815
[1479] Cai S *et al* 2019 Effective and efficient dropout for deep convolutional neural networks (arXiv:1904.03392)
[1480] Ghiasi G, Lin T-Y and Le Q V 2018 DropBlock: a regularization method for convolutional networks *Advances in Neural Information Processing Systems* pp 10727–37
[1481] Faramarzi M, Amini M, Badrinaaraayanan A, Verma V and Chandar S 2020 PatchUp: a regularization technique for convolutional neural networks (arXiv:2006.07794)
[1482] Kang G, Li J and Tao D 2017 Shakeout: a new approach to regularized deep neural network training *IEEE Trans. Pattern Anal. Mach. Intell.* **40** 1245–58
[1483] Kang G, Li J and Tao D 2016 Shakeout: a new regularized deep neural network training scheme *Proc. 30th Conf. on Artificial Intelligence* pp 1751–7
[1484] Zhou M *et al* 2019 Towards understanding the importance of noise in training neural networks (arXiv:1909.03172)
[1485] Graves A, Mohamed A-R and Hinton G 2013 Speech recognition with deep recurrent neural networks *2013 IEEE Int. Conf. on Acoustics, Speech and Signal Processing* (IEEE) pp 6645–9
[1486] Graves A 2011 Practical variational inference for neural networks *Advances in Neural Information Processing Systems* pp 2348–56
[1487] Sum J, Leung C-S and Ho K 2019 A limitation of gradient descent learning *IEEE Trans. Neural Netw. Learn. Syst.* **31** 2227–32
[1488] Holmstrom L and Koistinen P 1992 Using additive noise in back-propagation training *IEEE Trans. Neural Netw.* **3** 24–38
[1489] You Z, Ye J, Li K, Xu Z and Wang P 2019 Adversarial noise layer: regularize neural network by adding noise *2019 IEEE Int. Conf. on Image Processing (ICIP)* (IEEE) pp 909–13
[1490] Jenni S and Favaro P 2019 On stabilizing generative adversarial training with noise *Proc. Conf. on Computer Vision and Pattern Recognition* pp 12145–53
[1491] Sun Y, Tian Y, Xu Y and Li J 2019 Limited gradient descent: learning with noisy labels *IEEE Access* **7** 168296–306
[1492] Simsekli U, Sagun L and Gurbuzbalaban M 2019 A tail-index analysis of stochastic gradient noise in deep neural networks (arXiv:1901.06053)
[1493] Neelakantan A *et al* 2015 Adding gradient noise improves learning for very deep networks (arXiv:1511.06807)
[1494] Shorten C and Khoshgoftaar T M 2019 A survey on image data augmentation for deep learning *Journal of Big Data* **6** 60
[1495] Raileanu R, Goldstein M, Yarats D, Kostrikov I and Fergus R 2020 Automatic data augmentation for generalization in deep reinforcement learning (arXiv:2006.12862)
[1496] Antczak K On regularization properties of artificial datasets for deep learning (arXiv:1908.07005)
[1497] Ouali Y, Hudelot C and Tami M 2020 An overview of deep semi-supervised learning (arXiv:2006.05278)
[1498] Zhu J 2020 Semi-supervised learning: the case when unlabeled data is equally useful (arXiv:2005.11018)
[1499] Aitchison L 2020 A statistical theory of semi-supervised learning (arXiv:2008.05913)
[1500] Bagherzadeh J and Asil H 2019 A review of various semi-supervised learning models with a deep learning and memory approach *Iran J. Comput. Sci.* **2** 65–80
[1501] Rasmus A, Berglund M, Honkala M, Valpola H and Raiko T 2015 Semi-supervised learning with ladder networks *Advances in Neural Information Processing Systems* pp 3546–54
[1502] Lee D-H 2013 Pseudo-label: the simple and efficient semi-supervised learning method for deep neural networks *Workshop on Challenges in Representation Learning, ICML* vol 3
[1503] Sun S, Mao L, Dong Z and Wu L 2019 Multiview transfer learning and multitask learning *Multiview Machine Learning* (Berlin: Springer) pp 85–104
[1504] Ruder S 2017 An overview of multi-task learning in deep neural networks (arXiv:1706.05098)
[1505] Thung K-H and Wee C-Y 2018 A brief review on multi-task learning *Multimedia Tools Appl.* **77** 29705–25
[1506] Zhang Y and Yang Q 2017 A survey on multi-task learning (arXiv:1707.08114)
[1507] Caruana R 1997 Multitask learning *Mach. Learn.* **28** 41–75
[1508] Odena A, Olah C and Shlens J 2016 Conditional image synthesis with auxiliary classifier GANs (arXiv:1610.09585)
[1509] Shu R, Bui H and Ermon S 2017 AC-GAN learns a biased distribution *NIPS Workshop on Bayesian Deep Learning* vol 8
[1510] Gong M, Xu Y, Li C, Zhang K and Batmanghelich K 2019 Twin auxilary classifiers GAN *Advances in Neural Information Processing Systems* pp 1330–9
[1511] Han L, Stathopoulos A, Xue T and Metaxas D 2020 Unbiased auxiliary classifier GANs with MINE (arXiv:2006.07567)
[1512] Better Performance with the tf.data API 2020 TensorFlow Documentation (available at: www.tensorflow.org/guide/data_performance)
[1513] Li B, Wu F, Lim S-N, Belongie S and Weinberger K Q 2020 On feature normalization and data augmentation (arXiv:2002.11102)
[1514] Bhanja S and Das A 2018 Impact of data normalization on deep neural network for time series forecasting (arXiv:1812.05519)
[1515] van Hasselt H P, Guez A, Hessel M, Mnih V and Silver D 2016 Learning values across many orders of magnitude *Advances in Neural Information Processing Systems* pp 4287–95
[1516] Li M, Soltanolkotabi M and Oymak S 2020 Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks *Int. Conf. on Artificial Intelligence and Statistics* pp 4313–24
[1517] Flynn T, Yu K M, Malik A, D'Imperio N and Yoo S 2020 Bounding the expected run-time of nonconvex optimization with early stopping (arXiv:2002.08856)
[1518] Nagaraj D, Jain P and Netrapalli P 2019 SGD without replacement: sharper rates for general smooth convex functions *Int. Conf. on Machine Learning* pp 4703–11

[1519] Gürbüzbalaban M, Ozdaglar A and Parrilo P 2019 Why random reshuffling beats stochastic gradient descent *Math. Program.*
[1520] Haochen J and Sra S 2019 Random shuffling beats SGD after finite epochs *Int. Conf. on Machine Learning* pp 2624–33
[1521] Shamir O 2016 Without-replacement sampling for stochastic gradient methods *Advances in Neural Information Processing Systems* pp 46–54
[1522] Bottou L 2009 Curiously fast convergence of some stochastic gradient descent algorithms *Proc. Symp. on Learning and Data Science*
[1523] tf.data.Dataset 2020 TensorFlow Documentation (available at: www.tensorflow.org/api_docs/python/tf/data/Dataset)
[1524] Harrington P D B 2018 Multiple versus single set validation of multivariate models to avoid mistakes *Crit. Rev. Anal. Chem.* **48** 33–46
[1525] Breiman L 1996 Bagging predictors *Mach. Learn.* **24** 123–40
[1526] Breiman L 2001 Random forests *Mach. Learn.* **45** 5–32
[1527] Goel E, Abhilasha E, Goel E and Abhilasha E 2017 Random forest: a review *Int. J. Adv. Res. Comput. Sci. Softw. Eng.* **7** 251–7
[1528] Probst P, Wright M N and Boulesteix A-L 2019 Hyperparameters and tuning strategies for random forest *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **9** e1301
[1529] Xu Y and Goodacre R 2018 On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning *J. Anal. Test.* **2** 249–62
[1530] Guyon I 1997 A scaling law for the validation-set training-set size ratio *AT&T Bell Laboratories* **1**
[1531] Newman M E J 2005 Power laws, pareto distributions and Zipf's law *Contemp. Phys.* **46** 323–51
[1532] Opeyemi B 2019 Deployment of machine learning models demystified (part 1) Towards Data Science (available at: https://towardsdatascience.com/deployment-of-machine-learning-model-demystified-part-1-1181d91815d2)
[1533] Opeyemi B 2019 Deployment of machine learning model demystified (part 2) Medium (available at: https://medium.com/@opeyemibami/deployment-of-machine-learning-models-demystified-part-2-63eadaca1571)
[1534] Wu C-J *et al* 2019 Machine learning at facebook: understanding inference at the edge *2019 IEEE Int. Symp. on High Performance Computer Architecture (HPCA)* (IEEE) pp 331–44
[1535] Cai H, Gan C and Han S 2019 Once for all: train one network and specialize it for efficient deployment (arXiv:1908.09791)
[1536] Suresh A and Ganesh Kumar P 2020 Optimization of metascheduler for cloud machine learning services *Wirel. Pers. Commun.* **114** 367–88
[1537] Kumar Y, Kaul S and Sood K 2019 Effective use of the machine learning approaches on different clouds *Proc. Int. Conf. on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan* (Jaipur, India)
[1538] Dubois D J, Trubiani C and Casale G 2016 Model-driven application refactoring to minimize deployment costs in preemptible cloud resources *2016 IEEE 9th Int. Conf. on Cloud Computing (CLOUD)* (IEEE) pp 335–42
[1539] Oracle *et al* GraphPipe: machine learning model deployment made simple
[1540] FlatBuffers: Memory Efficient Serialization Library 2020 FlatBuffers documentation (available at: https://google.github.io/flatbuffers)
[1541] Blalock D, Ortiz J J G, Frankle J and Guttag J 2020 What is the state of neural network pruning? (arXiv:2003.03033)
[1542] Pasandi M M, Hajabdollahi M, Karimi N and Samavi S 2020 Modeling of pruning techniques for deep neural networks simplification (arXiv:2001.04062)
[1543] Wu H, Judd P, Zhang X, Isaev M and Micikevicius P 2020 Integer quantization for deep learning inference: principles and empirical evaluation (arXiv:2004.09602)
[1544] Nayak P, Zhang D and Chai S 2019 Bit efficient quantization for deep neural networks (arXiv:1910.04877)
[1545] Zhou Y, Moosavi-Dezfooli S-M, Cheung N-M and Frossard P 2017 Adaptive quantization for deep neural network (arXiv:1712.01048)
[1546] Yang J *et al* 2019 Quantization networks *Proc. Conf. on Computer Vision and Pattern Recognition* pp 7308–16
[1547] Zhuang B *et al* 2019 Effective training of convolutional neural networks with low-bitwidth weights and activations (arXiv:1908.04680)
[1548] Li H *et al* 2017 Training quantized nets: a deeper understanding *Advances in Neural Information Processing Systems* pp 5811–21
[1549] Wang S and Kanwar P 2019 BFloat16: the secret to high performance on cloud TPUs google cloud (available at: https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus)
[1550] Marco V S, Taylor B, Wang Z and Elkhatib Y 2020 Optimizing deep learning inference on embedded systems through adaptive model selection *ACM Trans. Embedded Comput. Syst. (TECS)* **19** 1–28
[1551] Jackson B 2020 How to optimize images for web and performance kinsta blog (available at: https://kinsta.com/blog/optimize-images-for-web)
[1552] Leventić H, Nenadić K, Galić I and Livada Č 2016 Compression parameters tuning for automatic image optimization in web applications *2016 Int. Symp. ELMAR* (IEEE) pp 181–4
[1553] Olah C, Mordvintsev A and Schubert L 2017 Feature visualization distill (available at: https://distill.pub/2017/feature-visualization)
[1554] Xie N, Ras G, van Gerven M and Doran D 2020 Explainable deep learning: a field guide for the uninitiated (arXiv:2004.14545)
[1555] Vilone G and Longo L 2020 Explainable artificial intelligence: a systematic review (arXiv:2006.00093)
[1556] Arrieta A B *et al* 2020 Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI *Inf. Fusion* **58** 82–115
[1557] Puiutta E and Veith E 2020 Explainable reinforcement learning: a survey (arXiv:2005.06247)
[1558] Gunning D and Aha D W 2019 DARPA's explainable artificial intelligence program *AI Mag.* **40** 44–58
[1559] Samek W and Müller K-R 2019 Towards Explainable Artificial Intelligence *Explainable AI: Interpreting, explaining and visualizing deep learning* (Berlin: Springer) pp 5–22
[1560] Hase P and Bansal M 2020 Evaluating explainable AI: which algorithmic explanations help users predict model behavior? (arXiv:2005.01831)
[1561] Ullah I *et al* 2020 A brief survey of visual saliency detection *Multimedia Tools Appl.* **79** 34605–45
[1562] Borji A, Cheng M-M, Hou Q, Jiang H and Li J 2019 Salient object detection: a survey *Comput. Visual Media* 1–34
[1563] Cong R *et al* 2018 Review of visual saliency detection with comprehensive information *IEEE Trans. Circuits Syst. Video Technol.* **29** 2941–59
[1564] Borji A, Cheng M-M, Jiang H and Li J 2015 Salient object detection: a benchmark *IEEE Trans. Image Process.* **24** 5706–22

[1565] Rebuffi S-A, Fong R, Ji X and Vedaldi A 2020 There and back again: revisiting backpropagation saliency methods *Proc. IEEE/Conf. on Computer Vision and Pattern Recognition* pp 8839–48
[1566] Wang Y, Su H, Zhang B and Hu X 2019 Learning reliable visual saliency for model explanations *IEEE Trans. Multimedia* **22** 1796–807
[1567] Kim B *et al* 2019 Why are saliency maps noisy? Cause of and solution to noisy saliency maps (arXiv:1902.04893)
[1568] Selvaraju R R *et al* 2017 Grad-CAM: visual explanations from deep networks via gradient-based localization *Proc. IEEE Int. Conf. on Computer Vision* pp 618–26
[1569] Morbidelli P, Carrera D, Rossi B, Fragneto P and Boracchi G 2020 Augmented Grad-CAM: heat-maps super resolution through augmentation *ICASSP 2020-2020 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 4067–71
[1570] Omeiza D, Speakman S, Cintas C and Weldermariam K 2019 Smooth grad-CAM++: an enhanced inference level visualization technique for deep convolutional neural network models (arXiv:1908.01224)
[1571] Chattopadhay A, Sarkar A, Howlader P and Balasubramanian V N 2018 Grad-Cam++: generalized gradient-based visual explanations for deep convolutional networks *2018 IEEE Conf. on Applications of Computer Vision (WACV)* (IEEE) pp 839–47
[1572] Patro B N, Lunayach M, Patel S and Namboodiri V P 2019 U-Cam: visual explanation using uncertainty based class activation maps *Proc. IEEE Int. Conf. on Computer Vision* pp 7444–53
[1573] Borji A 2019 Saliency prediction in the deep learning era: successes and limitations *IEEE Trans. Pattern Anal. Mach. Intell.*
[1574] Wang W *et al* 2019 Revisiting video saliency prediction in the deep learning era *IEEE Trans. Pattern Anal. Mach. Intell.*
[1575] Chen L, Chen J, Hajimirsadeghi H and Mori G 2020 Adapting grad-CAM for embedding networks *The IEEE Conf. on Applications of Computer Vision* pp 2794–803
[1576] Ramaswamy H G *et al* 2020 Ablation-CAM: visual explanations for deep convolutional network via gradient-free localization *The IEEE Conf. on Applications of Computer Vision* pp 983–91
[1577] Wang H *et al* 2020 Score-CAM: score-weighted visual explanations for convolutional neural networks *Proc. IEEE/Conf. on Computer Vision and Pattern Recognition Workshops* pp 24–5
[1578] Cancela B, Bolón-Canedo V, Alonso-Betanzos A and Gama J 2020 A scalable saliency-based feature selection method with instance-level information *Knowl.-Based Syst.* **192** 105326
[1579] Cheng M-M, Mitra N J, Huang X, Torr P H and Hu S-M 2014 Global contrast based salient region detection *IEEE Trans. Pattern Anal. Mach. Intell.* **37** 569–82
[1580] Nguyen A, Yosinski J and Clune J 2019 Understanding neural networks via feature visualization: a survey *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Berlin: Springer) pp 55–76
[1581] Xiao W and Kreiman G 2019 Gradient-free activation maximization for identifying effective stimuli (arXiv:1905.00378)
[1582] Erhan D, Bengio Y, Courville A and Vincent P 2009 Visualizing higher-layer features of a deep network *University of Montreal* **1341**
[1583] Mordvintsev A, Olah C and Tyka M 2015 Inceptionism: going deeper into neural networks google AI blog (available at: https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html)
[1584] Maaten L v d and Hinton G 2008 Visualizing data using t-SNE *J. Mach. Learn. Res.* **9** 2579–605
[1585] Wattenberg M, Viégas F and Johnson I 2016 How to use t-SNE effectively *Distill* **1** e2
[1586] Van Der Maaten L 2013 Barnes-Hut-SNE (arXiv:1301.3342)
[1587] Barnes J and Hut P 1986 A hierarchical $O(N\log N)$ force-calculation algorithm *Nature* **324** 446–9
[1588] Wang Z J *et al* 2020 CNN explainer: learning convolutional neural networks with interactive visualization (arXiv:2004.15004)
[1589] Wang Z J *et al* 2020 CNN 101: interactive visual learning for convolutional neural networks *Extended Abstracts of the 2020 Conf. on Human Factors in Computing Systems* pp 1–7
[1590] Kahng M, Thorat N, Chau D H P, Viégas F B and Wattenberg M 2018 GAN Lab: understanding complex deep generative models using interactive visual experimentation *IEEE Trans. Vis. Comput. Graphics* **25** 1–11
[1591] Gangavarapu T, Jaidhar C and Chanduka B 2020 Applicability of machine learning in spam and phishing email filtering: review and approaches *Artif. Intell. Rev.* **53** 5019–81
[1592] Dada E G *et al* 2019 Machine learning for email spam filtering: review, approaches and open research problems *Heliyon* **5** e01802
[1593] Bhuiyan H, Ashiquzzaman A, Juthi T I, Biswas S and Ara J 2018 A survey of existing e-mail spam filtering methods considering machine learning techniques *Glob. J. Comput. Sci. Technol.* **18**
[1594] Zhang J and Zeng W 2020 Mining scientific and technical literature: from knowledge extraction to summarization *Trends and Applications of Text Summarization Techniques* (IGI Global)
[1595] Dangovski R, Jing L, Nakov P, Tatalović M and Soljačić M 2019 Rotational unit of memory: a novel representation unit for rnns with scalable applications *Trans. Assoc. Comput. Linguist.* **7** 121–38
[1596] Scholarcy: The AI-Powered Article Summarizer 2020 (available at: www.scholarcy.com)
[1597] Romanov A, Lomotin K and Kozlova E 2019 Application of natural language processing algorithms to the task of automatic classification of russian scientific texts *Data Sci. J.* **18** 37
[1598] Gonçalves S, Cortez P and Moro S 2019 A deep learning classifier for sentence classification in biomedical and computer science abstracts *Neural Comput. Appl.* **32** 6793–807
[1599] Hughes M, Li I, Kotoulas S and Suzumura T 2017 Medical text classification using convolutional neural networks *Stud. Health Technol. Inf.* **235** 246–50
[1600] Liu J, Xu Y and Zhu Y 2019 Automated essay scoring based on two-stage learning (arXiv:1901.07744)
[1601] Dong F, Zhang Y and Yang J 2017 Attention-based recurrent convolutional neural network for automatic essay scoring *Proc. 21st Conf. on Computational Natural Language Learning (CoNLL 2017)* pp 153–62
[1602] Taghipour K and Ng H T 2016 A neural approach to automated essay scoring *Proc. 2016th Conf. on Empirical Methods in Natural Language Processing* pp 1882–91
[1603] Alikaniotis D, Yannakoudakis H and Rei M 2016 Automatic text scoring using neural networks (arXiv:1606.04289)
[1604] Foltýnek T, Meuschke N and Gipp B 2019 Academic plagiarism detection: a systematic literature review *ACM Comput. Surv. (CSUR)* **52** 1–42
[1605] Meuschke N, Stange V, Schubotz M, Kramer M and Gipp B 2019 Improving academic plagiarism detection for STEM documents by analyzing mathematical content and citations *2019 ACM/IEEE Conf. on Digital Libraries (JCDL)* (IEEE) pp 120–9
[1606] Ullah F, Wang J, Farhan M, Habib M and Khalid S 2018 Software plagiarism detection in multiprogramming languages using machine learning approach *Concurrency and Computation: Practice and Experience* p e5000

[1607] Lakkaraju H *et al* 2015 A machine learning framework to identify students at risk of adverse academic outcomes *Proc. 21th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* pp 1909–18
[1608] Foster D 2019 *Generative Deep Learning: Teaching Machines to Paint, Write, Compose and Play* (O'Reilly Media)
[1609] Zhan H, Dai L and Huang Z 2019 Deep learning in the field of art *Proc. 2019 Int. Conf. on Artificial Intelligence and Computer Science* pp 717–19
[1610] Dhariwal P *et al* 2020 Jukebox: a generative model for music (arXiv:2005.00341)
[1611] Briot J-P and Pachet F 2020 Deep learning for music generation: challenges and directions *Neural Comput. Appl.* **32** 981–93
[1612] Briot J-P, Hadjeres G and Pachet F-D 2020 *Deep Learning Techniques for Music Generation* (Berlin: Springer)
[1613] Brown T B *et al* 2020 Language models are few-shot learners (arXiv:2005.14165)
[1614] Radford A *et al* 2019 Better language models and their implications OpenAI blog (available at: https://openai.com/blog/better-language-models)
[1615] Chen H, Le T H M and Babar M A 2020 Deep learning for source code modeling and generation: models, applications and challenges *ACM Comput. Surv. (CSUR)* **53** 62
[1616] Allamanis M, Barr E T, Devanbu P and Sutton C 2018 A survey of machine learning for big code and naturalness *ACM Comput. Surv. (CSUR)* **51** 1–37
[1617] Autocompletion with deep learning 2019 TabNine Blog (available at: www.tabnine.com/blog/deep)
[1618] Svyatkovskiy A, Deng S K, Fu S and Sundaresan N 2020 IntelliCode compose: code generation using transformer (arXiv:2005.08025)
[1619] Hammad M, Babur O, Basit H A and Brand M V D 2020 DeepClone: modeling clones to generate code predictions (arXiv:2007.11671)
[1620] Schuster R, Song C, Tromer E and Shmatikov V 2020 You autocomplete me: poisoning vulnerabilities in neural code completion (arXiv:2007.02220)
[1621] Svyatkovskoy A *et al* 2020 Fast and memory-efficient neural code completion (arXiv:2004.13651)
[1622] Hellendoorn V J, Proksch S, Gall H C and Bacchelli A 2019 When code completion fails: a case study on real-world completions *2019 IEEE/ACM 41st Int. Conf. on Software Engineering (ICSE)* (IEEE) pp 960–70
[1623] Balog M, Gaunt A L, Brockschmidt M, Nowozin S and Tarlow D 2017 DeepCoder: learning to write programs *Int. Conf. on Learning Representations (ICLR 2017)* (OpenReview.net)
[1624] Murali V, Qi L, Chaudhuri S and Jermaine C 2018 Neural sketch learning for conditional program generation (arXiv:1703.05698)
[1625] Demir S, Mutlu U and Özdemir O 2019 Neural academic paper generation (arXiv:1912.01982)
[1626] SciNote 2020 Manuscript writer (available at: www.scinote.net/manuscript-writer)
[1627] Stribling J, Krohn M and Aguayo D 2005 SCIgen—an automatic CS paper generator (available at: https://pdos.csail.mit.edu/archive/scigen)
[1628] Raghu M and Schmidt E 2020 A survey of deep learning for scientific discovery (arXiv:2003.11755)
[1629] Kepner J, Cho K and Claffy K 2019 New phenomena in large-scale internet traffic (arXiv:1904.04396 [cs.NI])
[1630] Adekitan A I, Abolade J and Shobayo O 2019 Data mining approach for predicting the daily internet data traffic of a smart university *J. Big Data* **6** 11
[1631] Xu X, Wang J, Peng H and Wu R 2019 Prediction of academic performance associated with internet usage behaviors using machine learning algorithms *Comput. Human Behav.* **98** 166–73
[1632] Granger R 2020 Toward the quantification of cognition (arXiv:2008.05580)
[1633] Musk E *et al* 2019 An integrated brain-machine interface platform with thousands of channels *J. Med. Internet Res.* **21** e16194
[1634] Tshitoyan V *et al* 2019 Unsupervised word embeddings capture latent knowledge from materials science literature *Nature* **571** 95–8
[1635] Ruf J and Wang W 2020 Neural networks for option pricing and hedging: a literature review *J. Comput. Finance Forthcoming* **24**
[1636] Huang B, Huan Y, Xu L D, Zheng L and Zou Z 2019 Automated trading systems statistical and machine learning methods and hardware implementation: a survey *Enterprise Infor. Sys.* **13** 132–44
[1637] Raghavan M, Barocas S, Kleinberg J and Levy K 2020 Mitigating bias in algorithmic hiring: evaluating claims and practices *Proc. 2020th Conf. on Fairness, Accountability and Transparency* pp 469–81
[1638] Mahmoud A A, Shawabkeh T A, Salameh W A and Al Amro I 2019 Performance predicting in hiring process and performance appraisals using machine learning *2019 10th Int. Conf. on Information and Communication Systems (ICICS)* (IEEE) pp 110–15
[1639] Raub M 2018 Bots, bias and big data: artificial intelligence, algorithmic bias and disparate impact liability in hiring practices *Ark. Law Rev.* **71** 529
[1640] Newman N 2017 Reengineering workplace bargaining: how big data drives lower wages and how reframing labor law can restore information equality in the workplace *Univ. Cincinnati Law Rev.* **85** 693
[1641] Price W and Nicholson I G 2019 *Berkeley Technol. Law J.* **34** 1
[1642] Zhuang H and Acuna D E 2019 The effect of novelty on the future impact of scientific grants (arXiv:1911.02712)
[1643] Zhang W E, Sheng Q Z, Alhazmi A and Li C 2020 Adversarial attacks on deep-learning models in natural language processing: a survey *ACM Trans. Intell. Sys. Technol. (TIST)* **11** 1–41
[1644] Ma X *et al* 2020 Understanding adversarial attacks on deep learning based medical image analysis systems *Pattern Recognit.* **110** 107332
[1645] Yuan X, He P, Zhu Q and Li X 2019 Adversarial examples: attacks and defenses for deep learning *IEEE Trans. Neural Netw. Learn. Sys.* **30** 2805–24
[1646] Akhtar N and Mian A 2018 Threat of adversarial attacks on deep learning in computer vision: a survey *IEEE Access* **6** 14410–30
[1647] Goodfellow I J, Shlens J and Szegedy C 2014 Explaining and harnessing adversarial examples (arXiv:1412.6572)
[1648] Wen Y, Li S and Jia K 2019 Towards understanding the regularization of adversarial robustness on neural networks *OpenReview.net*
[1649] Lecuyer M, Atlidakis V, Geambasu R, Hsu D and Jana S 2019 Certified robustness to adversarial examples with differential privacy *2019 Symp. on Security and Privacy (SP)* (IEEE) pp 656–72
[1650] Li Y *et al* 2018 Optimal transport classifier: defending against adversarial attacks by regularized deep embedding (arXiv:1811.07950)
[1651] Xie C *et al* 2020 Adversarial examples improve image recognition *Proc. IEEE/Conf. on Computer Vision and Pattern Recognition* pp 819–28

[1652] Deniz O, Pedraza A, Vallez N, Salido J and Bueno G 2020 Robustness to adversarial examples can be improved with overfitting *Int. J. Mach. Learn. Cybern.* **11** 935–44
[1653] Kinoshita Y and Kiya H 2020 Fixed smooth convolutional layer for avoiding checkerboard artifacts in CNNs *ICASSP 2020–2020 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE) pp 3712–16
[1654] Xiao H, Rasul K and Vollgraf R 2017 Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms (arXiv:1708.07747)