VIEW **Communicated by Ben Tsuda**

Understanding and Applying Deep Learning

Richard Lippmann
richardp.lippmann@gmail.com
IEEE Life Fellow; Wayland, MA 01778, U.S.A.

The past 10 years have witnessed an explosion in deep learning neural network model development. The most common perceptual models with vision, speech, and text inputs are not general-purpose AI systems but tools. They automatically extract clues from inputs and compute probabilities of class labels. Successful applications require representative training data, an understanding of the limitations and capabilities of deep learning, and careful attention to a complex development process. The goal of this view is to foster an intuitive understanding of convolutional network deep learning models and how to use them with the goal of engaging a wider creative community. A focus is to make it possible for experts in areas such as health, education, poverty, and agriculture to understand the process of deep learning model development so they can help transition effective solutions to practice.

1 Introduction

Recent successes in artificial intelligence (AI) rely on learning systems called neural network models. This name originated because these models use many simple computation elements or nodes operating in parallel and arranged in layers reminiscent of biological neural networks. Nodes are connected via weights whose settings are learned during training. Modern "deep learning" neural network models demonstrate improved performance on many tasks, including locating and labeling objects in images, speech recognition, detecting fraudulent financial transactions, transcribing speech, analyzing text queries, medical diagnostics, and drug discovery (LeCun, Bengio, & Hinton, 2015; Vaswani et al., 2017).

Neural network capabilities have grown dramatically. A 1987 review of neural networks included only six common models and a few applications (Lippmann, 1987). A recent online review (Fridman, 2020) and more than 40 workshops in the important NeurIPS neural network research conference include more than 100 applications and approaches.[1] Deep learning using

[1] NeurIPS Conference information is available at https://neurips.cc/virtual/2021/index.html.

Neural Computation **35, 287–308** (2023)
https://doi.org/10.1162/neco_a_01518
© 2022 Massachusetts Institute of Technology

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

multilayer neural networks was known as an approach in 1987, but modern advances are founded on massive amounts of data, vast computation resources, and new large architectures sometimes containing as many as billions of settings learned from data.

Early research on neural networks explored multilayer perceptrons (MLPs) where all nodes in a layer are connected to every node in the next layer and computations feed forward from lower to upper layers (Lippmann, 1987). These fully connected systems provided good performance on some classification and prediction tasks, but required many trained weights or settings and were often too computationally expensive for tasks such as image classification. Convolutional networks (ConvNets, or CNNs) are feedforward networks that were initially developed for image classification and other vision tasks (LeCun et al., 2015). They include many fewer trainable settings than MLPs and are relatively invariant to the position of an object in an input image. Convolutional networks are now the dominant approach for image classification, object identification in images, face recognition, extracting text from images, preprocessing speech in speech recognizers, and classification tasks in many domains. They often provide substantial improvement in performance over existing state-of-the-art systems. Real-time implementations can be found in cameras and smart phones for face detection, face recognition, and limited-vocabulary offline speech recognition.

Recurrent networks were developed to process input tokens with no predefined length such as sequences of words for speech recognition or language translation and sequences of video frames to determine when a person performs a specific action (Hochreiter & Schmidhuber, 1997). These networks contain internal memory that holds information extracted from past tokens and is used in addition to the most recent input to produce outputs. More recently, complex feedforward networks called transformers have been developed to analyze sets of input tokens such as sentences or paragraphs of words for tasks such as language translation, text topic classification, question answering, and predicting the next word or words in a sentence (Vaswani et al., 2017; Brown et al., 2020). Transformers use a type of training called attention to determine those node outputs from one layer to weight most strongly when forming inputs for each node in the next layer. These networks all use supervised training where class labels of at least some training examples are provided. Generative adversarial networks (GANs) are complex networks that use unsupervised training and do not require labels for training (Goodfellow et al., 2014). GANs can synthesize examples that are similar to training data but are not in the training data. Systems have been developed to synthesize high-resolution novel images of faces, help edit images, and generate superresolution versions of blurry images (Creswell et al., 2017). Examples of artificially generated faces and chemicals generated by GANs (Karras et al., 2020) are available at https://thispersondoesnotexist.com and https://thischemicaldoesnotexist.com.

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

This view focuses on convolutional networks because they have been transformational in vision, health, drug design, security, speech, vehicle safety, and many other application areas (LeCun et al., 2015). They are also widely used and easy to apply to new application areas. Finally, understanding how they work illustrates both the strength and weaknesses of not only convolutional networks but of many other deep neural networks. The goal of this view is to provide an intuitive understanding of convolutional networks that encourages a wider creative community to use them. This includes describing what they do, how they work, the overall development process they require, and benefits and limitations.

This view was motivated by three observations. First, many people vastly overrate capabilities of deep learning AI as confirmed by responses to a recent AI literacy quiz (DeCario & Etzioni, 2021). In addition, terms of "superhuman" and "better than human" performance on specific vision, speech, and text data sets are used even when performance is not quite even normal-human. Many "superhuman" models that work well on simple data sets are fragile and fail dramatically with small changes in the data that humans easily handle. As part of this problem, some literature overpersonifies AI systems. One recent book states that a research group "invited AI to participate in its process" (Kissinger, Schmidt, & Huttenlocher, 2021, p. 9). This sets the stage for failure by vastly oversimplifying the importance and difficulty of collecting data, training, and applying deep learning.

A second motivation for this view was that most people do not understand the statistical nature of machine learning. It would be exciting to have a reasoning system that can understand novel new situations and explain how it comes to conclusions. Unfortunately, deep learning systems are good at extracting clues to classify inputs but poor at creating a deep understanding of the world represented by those images. Deep models are brittle, entirely dependent on training data, and often overconfident when making wrong decisions, and their actions are difficult to explain.

A final motivation for this view was to better democratize the use of deep learning systems and encourage a larger community to think about using AI for good. Currently, large online corporations have been the primary beneficiaries of deep learning. They have developed AI approaches to improve Internet search, select social media content that keeps users engaged, and present advertisements tuned to individual users. The more people who understand how deep learning works, the more ideas that might surface to generate new applications. Some of these could improve everyday living and work processes or focus on poverty, hunger, health, education, or one of the 13 other United Nations 2030 Sustainable Development Goals (Vinuesa et al., 2020). These types of applications are made more feasible by the steadily improving capability of deep learning models to run on small devices, including smartphones (Ignatov et al., 2018; Wang et al., 2020) and development platforms that require little programming knowledge (Google, 2022; Microsoft, 2022).

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

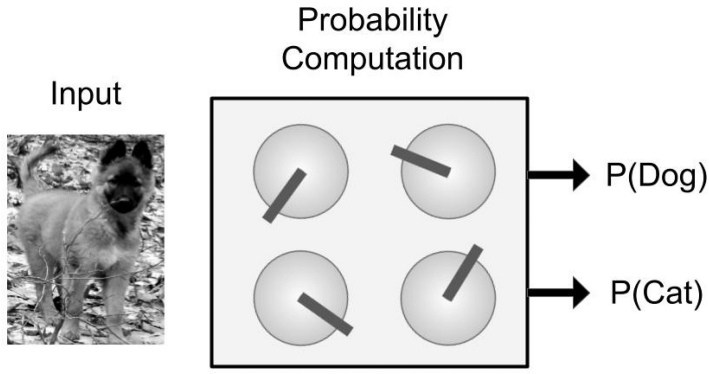

Figure 1: A deep learning model with knobs representing adjustable settings. Outputs are probabilities of an input image containing a dog or cat.

The remainder of this view focuses on convolutional network deep learning systems that are fed some type of input and help humans make decisions about that input. These include some of the most widely applied vision, speech, and text analysis models. New terms commonly used in the deep learning field are italicized when first presented to gently introduce important terminology.

2 Deep Learning Models Are Probability Computing Tools

Deep learning models, including multilayer perceptrons, convolutional networks, recurrent networks, and transformers, are often used as tools that help us decide which preselected *class label* best describes a given input. A model is like a spreadsheet where you copy in data that represent an input, and there are as many outputs as there are classes for possible inputs. Internally, the input data are fed through a fixed computation or function, and each output represents a guess at the probability that the input corresponds to one class. Probabilities are numbers between 0.0 (not present) and 1.0 (certainly present) that express how likely the label applies to the input. The only difference between a deep learning model and a spreadsheet is that the settings are trained using data instead of adjusted by hand.

A simple deep learning model that analyzes images has knobs representing adjustable parameters or settings and has two outputs, as shown in Figure 1. Outputs represent probabilities that the input image contains a dog or a cat. In other similar classifiers, the input could be an x-ray image and a patient's medical history and the output could be the probability of cancer. In more complex applications, there may be hundreds of outputs representing species of insects that might be infecting a crop or tens of thousands of outputs corresponding to all the words in a dictionary. There also

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

might be more complex inputs' including text, acoustic speech waveforms, and numerical data.

Outputs of neural net models are produced using a complex but fixed nonlinear function created during training by varying adjustable parameters to obtain good performance on training data. This function could be computed on something as small as a smartwatch using thousands of settings for simple applications, such as detecting when a short phrase is spoken to wake up a device (Chen & Ran, 2019), or it could be computed on a smartphone using a few million settings to recognize objects in images from a built-in camera (Howard et al., 2019). Alternatively, it may require large, centralized computer servers in the cloud when there are many millions of settings as in large image classification models (Tan & Le, 2019) or billions of settings as for some large recent language models used for text completion or constrained question answering (Brown et al., 2020).

Deep learning models do not have a high-level understanding of input classes. They do not understand what cancer is, how dangerous it is, or how it occurs and is cured. They do not need to understand why earthquakes, thunderstorms, or floods are important. They are simply trained to compute probabilities based on clues extracted from input data. It may seem odd that models can calculate probabilities from inputs, but this is something we do all the time. By looking at cars traveling on a road, we infer the probability of being safe when we pull our car into a stream of traffic. By reading the temperature of our children on a thermometer, we determine if they are sick. By looking at a garden, we can determine if it needs watering.

Deep learning models do not make decisions. It is up to users to determine how to use the probabilities provided. In simple applications, such as identifying your pet cat in a collection of images, you could simply say the cat is in an image if a cat-detector model produces a probability above 0.5. For more complex decisions, like deciding if there is a pedestrian in the road for autonomous driving or whether a person has cancer, these probabilities are one piece of information to take into consideration.

3 How Deep Learning Convolutional Vision Models Work

Although we see the world in 3D, the most common and successful deep learning vision models, called *convolutional nets*, compute probabilities from flat images (LeCun et al., 2015). You can understand how they work by thinking about how we solve jigsaw puzzles. Jigsaw puzzles are already broken up into small pieces. We pick up an individual puzzle piece, compare it to a full picture of the completed puzzle on the box cover, and try to find a texture, edge, color, or feature that the piece matches. We also try to combine the piece with other similar pieces to match larger patches of the full picture.

Instead of matching puzzle pieces to a full image, deep learning models first break the input up into small pieces or patches. They then compare each

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

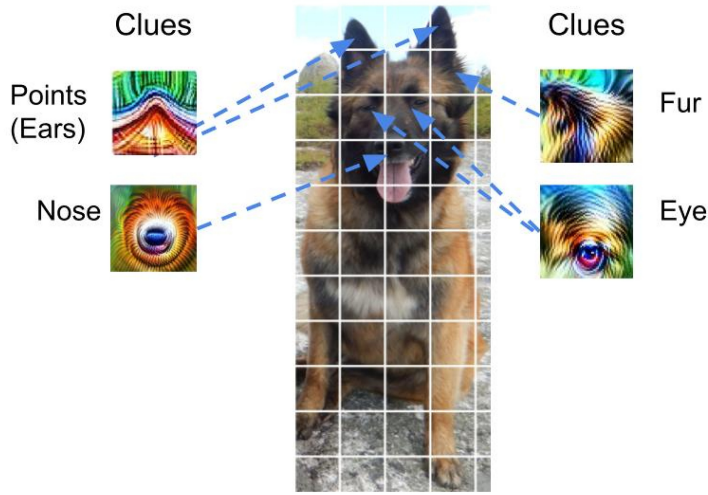

Figure 2: Patches in an input image are analyzed to determine if they contain clues to the image class. In this example, clues are found in different patches that contain evidence for points (ears), fur, two eyes, and a nose. These increase the output probability that the input contains a dog. Every patch is analyzed to determine if it contains any clue. Images of clues are from Olah et al. (2020).

patch to clues for the different classes of input images. The major advantage of deep learning models is that they automatically learn these clues during training. These might be as simple as edges at different orientations, unique colors or textures, or small curves or points. Clues are like highly identifiable puzzle pieces that are special for each class. During testing, the strength of matches to clues determines the output probability for each class.

Clues are stored and matches are computed in structures that will be called *filters* but also called *convolutional filters* or *matched filters*. At the lowest level, inputs to filters are numbers that represent the red, green, or blue (RGB) color values used in cameras and displays to determine the color of each small dot or pixel in an image patch. A filter operates on all pixels in a patch by multiplying the number for each pixel by a separate *weight* and summing all the products to produce an output. Weights are the settings of a deep learning model.

At higher layers, filters operate on the outputs of lower-layer filters and can create clues from larger areas of the input image. These could be lines, curves, light or color gradients, dark spots, pointy shapes, curved regions, textures, or even a face prototype that contains two dark spots above a central dark region as described in Olah et al. (2020). Figure 2 shows an input image of a dog broken into intermediate-sized patches. Four clues are shown using stylized images of input patches that best match each clue.

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

These clues represent the behavior of only a few thousand filters in a large pretrained model. The images are from Olah et al. (2020), who generated and hand-selected them to explain how convolutional models work. Arrows show a few of the patches that strongly match clues. For example, the filter that detects peaks matches the tips of a dog's ears, and the more complex filter that detects a dark spot matches the dog's nose. These clues all are associated with the class "dog" and cause the output probability for that class to increase. There may be thousands of other clue detectors in a deep learning model for different classes focusing on small features that distinguish among classes. Every filter is applied to every patch in the image to determine how closely a patch matches its clue. Probabilities are calculated by combining the outputs of high-level clue detectors.

The clues shown in Figure 2 were learned by training a deep learning convolutional network using 1.2 million images assigned to 1000 classes from a data set called ImageNet (Russakovsky et al., 2015). Roughly 12% of these classes are pictures of dogs from different breeds, so it is not surprising that there are clues for the dog shown. Over the years, however, it has been found that as long as there are enough training data, models learn similar low-level clues (Olah et al., 2020).

Instead of matching each clue to every input patch, one at a time in sequence, deep learning models are structured so that matching can be performed in parallel. Training large deep learning models was made feasible by *graphical processing units* (*GPUs*) that could efficiently perform these parallel computations. Applying trained models to images also requires computing filter outputs. Many modern cell phones already include hardware that provides this capability and can process multiple images per second (Ignatov et al., 2018).

A stylized diagram of a convolutional net is shown in Figure 3. A single input that could be an image, spoken word or phrase, medical record data, or a word of text is first converted to numbers that are fed into the lowest layer on the left. This conversion is sometimes called *embedding*, and details depend on the type of input. As noted above, these numbers are pixel color values for images. Numbers are processed by input linear filters. The tips of each of the triangles shown represent computing elements or nodes that compute the sum of products between weights and inputs of one filter. Although it seems too simple, each of these weighted sums determines how well a patch of input matches a clue. For spoken words, one low-level filter might extract high-frequency energy present in the sound for "s" as a clue for discriminating between *stop* and *go*. As noted above, for images, low-level filters might be sensitive to specific colors, edges of different orientation, low-level shapes, or textures. There may be fewer than 100 different clues in the input level that are matched to hundreds or thousands of input patches.

Linear processing is followed by nonlinear processing that allows the overall model to compute complex functions. In addition, specialized

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

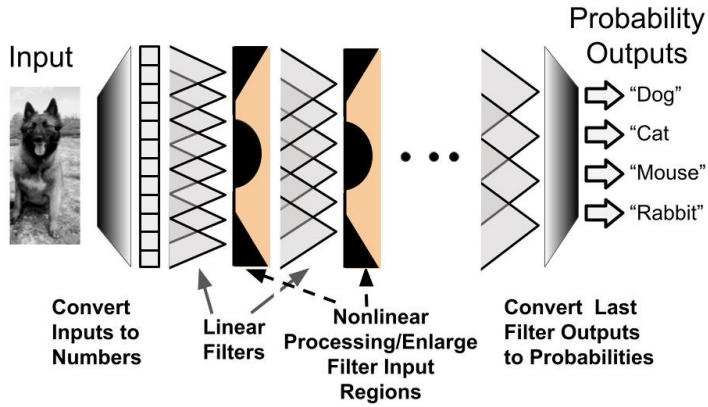

Figure 3: A deep learning convolutional neural net with multiple layers where the input is an image and outputs indicate label probabilities.

computations enlarge the extent of the input processed by upper-level filters. This makes upper filters able to combine clues from a larger part of the input image. These models are called deep not because they have deep understanding but because they contain many layers. Outputs after a final layer are processed to produce probabilities. There may be only a few layers to hundreds of layers and many thousands of filters extracting a wide variety of clues. Excellent visualizations of a wide variety of deep convolutional models are available at OpenAI (2022a). They illustrate the complex structure of many models, let you visualize clues at different levels, and display example input images that best match clues.

The structure of convolutional networks makes them relatively insensitive to where an object occurs in the input. Because clues are computed on all patches of an image, output probabilities change little as the image is moved right and left or up and down until it falls outside the input image. Network performance may still, however, degrade if the size of an object changes, the viewpoint or angle of an object changes, or the background changes.

As noted, there are other deep learning models. The most important is called a *transformer*. Instead of processing a single input image, word, or speech segment at a time as in convolutional networks, it processes a collection of inputs such as words in a paragraph of text and produces a collection of probabilities at the output (Vaswani et al., 2017). Transformers were initially used for natural language applications such as language translation and text completion, but they are now being applied successfully to image classification and other applications. Transformers also use linear filters and nonlinearities arranged in multiple layers. When used for processing text, clues are formed from specific sequences of words, while when used for

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

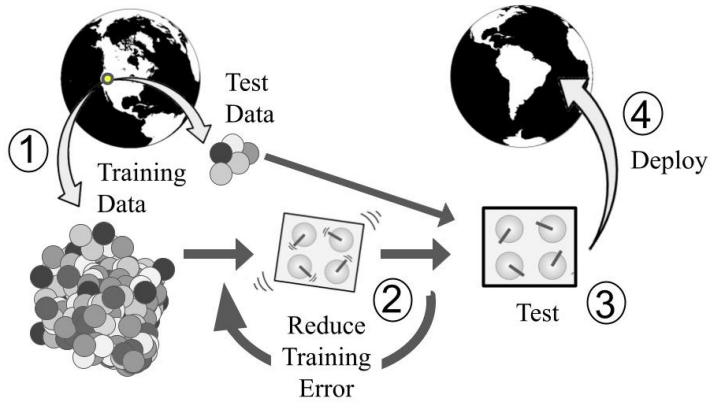

Figure 4: Machine learning application developers follow a four-step program. First, data are collected from a specific location and time and split into training and testing data. Second, a learning model's settings are adjusted using training data until its error rate is low enough. Third, the trained model is tested on test data to make sure it performs well on this previously unseen data. Finally, the trained model is deployed somewhere, often nowhere near where training data were collected and well after those data were collected.

images, clues are formed from image patches. A major difference in transformers is that linear filters are applied across members of the input collection using a process called *attention*. This can form new clues that combine information from items in different locations in an image or that occur in the past or future for items in a temporal sequence, as in speech recognition or text processing. Transformers currently provide the best performance when extremely large training data sets are available to train equally large models (Brown et al., 2020; Devlin, Chang, Kee, & Toutanova, 2019; Dai, Liu, Le, & Tan, 2021).

4 The Deep Learning Development Process

Developing machine learning applications requires a focus on data, how those data interact with deep learning models, and how to ensure good performance during deployment. The best way to keep the development process on track and focused is for someone who has a thorough understanding of a problem to participate in the four steps of deep learning development shown in Figure 4.

The four-step approach starts when someone comes up with an idea about how deep learning can address an important problem. The first step is collecting data from a specific location and time that characterize the problem. In Figure 4, data come from California in the United States. This could

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

be images, text, speech, graphs such as molecular structures, a variety of unstructured data such as patient records, or partly structured data such as spreadsheets. All data have to be collected, cleaned, and labeled using a predetermined set of classes.

The second step of development in Figure 4 begins with splitting data into two parts. *Training data* are used to train models until error on this data is low. *Test data* are tucked away safely without looking at them until training is over to make sure the model is not so *overtrained* and tuned to the training data that it no longer works well on different real-world data. As noted above, the type of training used for most models is called *supervised training* because labels, for example, in the training data are required to supervise or guide the training process. The main procedure is to start with a model that has randomized settings. A few training examples are selected and passed through the current model computing a *cost* using example labels that estimates how well model outputs represent actual probabilities. Model settings are adapted after every few examples in a manner that reduces the cost over time. This process continues, possibly going through all examples multiple times, until estimated performance no longer improves, as described below.

The third step of development is to test the trained model on previously unseen test data. This is an attempt to determine the expected performance of the model or *generalization* on new, unseen data. If the performance on test data is not good enough for this application, then more data, different data, or a different model might be necessary to address this problem. Alternatively, it could be that it is impossible to obtain better performance with the information available in the data provided.

Finally, if the test data error is low enough, the model is deployed. In Figure 4 this occurs in South America, even though it was developed in the United States. Although time has passed and the model is deployed far from where the training data were gathered, there is always hope that performance might actually be similar to that measured with test data. In practice, performance almost always drops substantially in the first live test of a model. More rounds of data collection and development are often required to improve performance on current real-world data collected where the model is deployed.

4.1 The Importance of Data. A common misunderstanding is that training data should not be required to solve a new problem because deep learning is a capable, advanced AI that can be applied to any situation. Nothing can be further than the truth. Data collection is the most important step of deep learning model development, and providing sufficient good data is the surest approach to good performance. It is not uncommon to spend 80% or more of your effort on obtaining, cleaning, labeling, and organizing data for training and testing in a new area. It is impossible to overstate how

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

many ways data can be messy, noisy, and corrupted and thus difficult to clean and use for training.

Data could be collected by a machine learning developer or a local expert like a doctor, banker, or farmer who is interested in a new application. An indication of the importance of data is a new track in the NeurIPS 2021 conference focusing on data sets and benchmarks (Vanschoren & Yeung, 2021) and a workshop on Data-Centric AI focusing on automating the work involved in gathering, cleaning, preprocessing, and labeling data.[2] Labeling each example is especially important because the accuracy of machine learning model training and evaluation depends on correct labeling. It is also important to make sure data include all important classes and examples of their everyday variability.

The long lifetime of many data sets is an indication of the importance of data and how difficult they are to collect. For example, the ImageNet data set contains roughly 1.2 million training images and 100,000 test images. It was created in 2010 and enabled the initial development of deep learning vision models in 2012 (Russakovsky et al., 2015). This data set is still used to improve and develop new models after 12 years. The current highest-performing vision and natural language models owe much of their performance to custom curated data sets containing billions of images or words (Brown et al., 2020; Dai et al., 2021).

4.2 An Overview of Training. Just as the original data are split into training and test data, examples set aside for training are normally split into two parts. The larger part is used to adjust weights, but *validation data* are set apart and used to measure *validation error* as training progresses. Training is stopped if the validation error increases, indicating the model is *overfitting* to unimportant details of the training data and may not perform well on other unseen data. A general overview of how training proceeds is shown in Figure 5.

Before training starts, model weights are set to random values within specified ranges. The first step of training is to select a few training examples called a *batch*. The second step is to process these examples and produce probability outputs from the current model. A *cost* is computed by comparing probability outputs to the correct labels. The cost computation uses a *cost function* that, if low enough, indicates the model is correctly estimating probabilities. The cost for one example is lowest when the highest output is for the example's class.

The goal of training is to reduce this model error over time but not to reduce it so much that the model overfits the training data. This is part of the art of training deep learning models. A common misunderstanding is

[2] The NeurIPS data centric workshop is described at https://datacentricai.org/neurips21/.

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

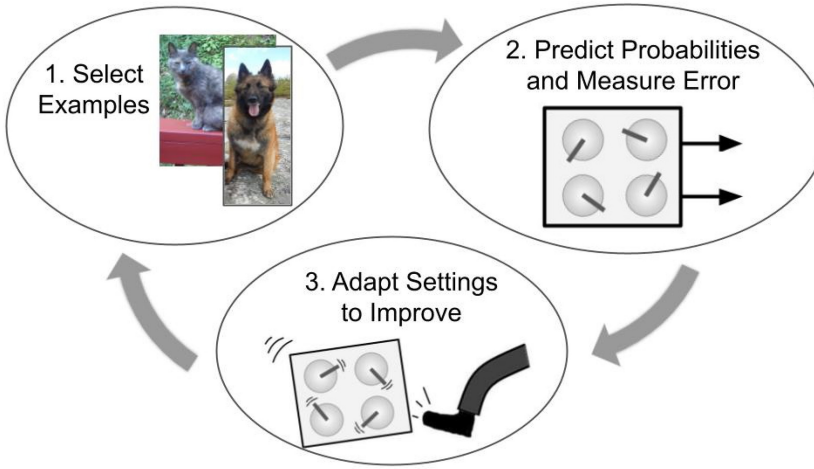

Figure 5: Training a deep learning model involves (1) repeatedly selecting a few training examples, (2) using the model to predict output probabilities and comparing them to the known input labels, and (3) adapting model settings to improve performance the next time those examples are presented.

that after training, networks produce 1.0 at the output corresponding to the current input label and 0.0 for other outputs. As noted above, outputs are probabilities, and since classes can almost never be perfectly distinguished from each other, probabilities are usually between 0.0 and 1.0.

After computing the cost for one batch of training examples, model settings are adjusted in a way that reduces the error the next time these examples are presented. Each setting could be moved back and forth to find the direction and magnitude that reduce the cost for these examples, just as you might tune an analog radio, but this would take too long. You could also simply kick or randomly move all settings until you stumble on a setting that reduces the error, but this would also take too long. Instead, calculus is used to compute derivatives that determine how each setting affects the cost. Settings are changed in a direction and magnitude that depend on the strength and direction of their individual effect. Settings that have a strong influence on the output cost are changed a great deal, while settings with a weak influence are changed little. The name given to the process of determining how much to change settings is *backpropagatio*n because derivatives are computed first at the network output and then propagate to earlier network levels. The amount of correction is usually multiplied by a *step size* or *learning rate* that can be set manually or automatically and is frequently reduced as training proceeds.

This process continues by running through all training examples. A single pass through all training examples is called an *epoch*, and it may take

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

many epochs to fully train a model. This overall approach to training is called *stochastic gradient descent*, and it can take minutes to days to complete, depending on the size of the data and the amount of computation available.

A simpler approach to training, called *transfer learning*, can greatly reduce the amount of labeled training data required. This involves obtaining a pretrained model and *tuning* that model using the data collected and labeled for your specific application. Many pretrained models called *foundation models* are available for this purpose (Bommasani et al., 2021). For example, in many medical applications, a convolution net can be pretrained on the ImageNet data set, and then smaller numbers of image data can be used to tune the model for a specialized medical imaging application (Greenspan, van Ginneken, & Summers, 2016). This still requires custom well-labeled data for an application but not as much as would be required to completely train a model. The same is true for natural language models designed for problems such as answering text queries. These models can be pretrained on billions of words of text and then adapted using question-and-answer pairs for a specific application (Brown et al., 2020; Devlin et al., 2019).

Another new approach, *self-supervised learning*, has been developed to reduce the number of labeled training examples required. With self-supervised training, large numbers of unlabeled data are first used to train a model. This model is then tuned using smaller numbers of training data as with transfer learning. For example, large natural language models are being trained on text from the Internet by learning to either predict the next word given prior words or how to fill in missing words given surrounding words (Brown et al., 2020; Devlin et al., 2019). After training, the core of these models can be tuned with a few labeled data to perform a natural language task such as question answering. Other self-supervised approaches have been successful for image classification and speech recognition (Chen, Xie, & He, 2021; Chen et al., 2021).

4.3 Why Errors Increase When a Model Is Deployed. Performance drops when a model is deployed for many reasons—for example, test data differ from training data, definitions of classes might be different between training and deployment, data might be collected differently, or labels might be incorrect in training data.

There are many examples of this effect. Word error rates for commercial speech to text systems for African American speakers averaged an unusable high 35% for conversational speech (Koenecke, Nam, Lake, & Goel, 2020) compared to reported low word error rates of less than 4% for deep learning speech models tested on clean speech (Chen et al., 2021). A recent clinical evaluation was performed in Thailand on a deep learning system developed to detect retinopathy from images of a person's retina. This system was trained using more than 128,000 labeled retina images and was

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

found to be comparable to trained ophthalmologists (Gulshan, Peng, & Coram, 2016). In the clinic, however, it could not be applied to more than 20% of the images. The system was trained only on high-quality images. It refused to rate low-quality images but *rejected* them even when it was clear to nurses that patients were free of the eye disease. Rejection can be an effective approach to ensuring that performance of deep learning classifiers remains high, but the clinical staff in this evaluation were unprepared for the unexpected workload. Finally, a convolutional net trained on chest x-ray images was shown to diagnose pneumonia as well as radiologists did at one hospital (Rajpurkar et al., 2017). When the same system was tested at a nearby hospital, performance dropped substantially because a slightly different machine and imaging protocol was used.

These degradations can only be addressed by assuming deployment is part of the overall development cycle. After deployment, more cycles of data collection and model improvement should be expected and built into the overall model development process. Ideally, this would include periodic or continuous performance monitoring and automated data collection to support retraining.

5 A Recent Success Story in Drug Discovery

Researchers at MIT recently used deep learning to help discover a new antibiotic as described in Stokes et al. (2020) and Yang et al. (2019). This is the research mentioned in Kissinger et al. (2021), where it says that the research group at MIT "invited AI to participate in its process." It is informative to examine how multiple experts from many fields cooperated to make this a success and to understand the role deep learning played.

Performing experiments to determine the antibiotic activity of molecules was the first step of this research. Laboratory analyses were performed on 2335 molecules to find those that inhibit *Escherichia coli* growth. Only 120 molecules were found to be 80% or more effective in inhibiting growth and were labeled as "growth inhibiting." The remaining molecules were labeled as "not growth inhibiting." This provided 2335 examples of labeled training data. These were very *unbalanced* data because only roughly 5% of the examples could be used to learn clues about how to identify antibiotically active molecules.

The second step of this study (step 2 in Figure 4) was to create numerical inputs to train classifiers. This was complex because inputs are molecules of different sizes and shapes. Information concerning the molecular structure of each molecule was summarized using a type of deep network called a *graph neural network.* It includes multiple levels of processing with nonlinearities as in a convolutional deep network but operates on graphs instead of images (Yang et al., 2019). The numerical representation created from the graph neural network was supplemented by adding other molecular information. These combined data were used to train an ensemble of binary

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

machine learning classifiers that worked well on training data to discriminate between molecules that did and did not inhibit *E. coli* growth.

The first deployment of this screening classifier (step 4 in Figure 4) involved using it to sort the antibiotic potential of 6111 molecules from a repository called the Drug Repurposing Hub. The 99 molecules with the highest "growth-inhibiting" probability outputs were selected for laboratory testing. Testing revealed that roughly half of these (51 of 99) significantly inhibited *E. coli* growth. The *sensitivity* of a network such as this is the percentage of the top-scoring selected test examples that are actually the desired class. This was only roughly 50%, which would have been unacceptable in medical diagnostic, image, or speech applications. In this application, however, this was sufficient because it was relatively inexpensive to perform 99 additional lab experiments to analyze the top 99 molecules. The 51 molecules identified were put through a second pass of screening based on factors related to how rapidly they could be approved as a drug, their potential toxicity, and structural similarity to molecules in the primary training data set. One molecule was selected from this second screening pass. This molecule was almost missed. It was prioritized as number 89 in the top 99 molecules, and its probability of being effective was estimated to be only 0.33. It was lab-tested more thoroughly and found to be a broad-spectrum bactericidal antibiotic. Renamed halicin, it is being more thoroughly tested (Booq, Tawfik, Alfassam, Alfahad, & Essam, 2021).

This application of deep networks was complex but worked because experts in multiple fields were available for all steps of the process. They were there to perform experiments necessary to label training data, format input data, train a classifier, select and screen candidate drugs, perform experiments to determine whether screened candidates inhibit *E. coli* growth, and finally select the most likely positive candidate by hand using factors not taken into account by the screening network. None of these steps was easy. Although the screening network did not work that well, it was accurate enough because a second stage of laboratory experiments and manual screening identified an effective antibiotic.

6 Setting Expectations

Before even thinking about applying deep learning models, it is important to understand their strengths and weaknesses. These models work well with large numbers of training data, but they can be fragile and have other weaknesses. This section is a reminder of these issues.

6.1 Sometimes a Deep Learning Model Won't Make a Difference. Relatively simple classifiers work well for a surprisingly high fraction of real-world problems and data sets (Holte, 1993). Deep learning models are required only when large amounts of data are available to support training of such complex models. If there are only small numbers of data or if the

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

data can be easily modeled and explained, then a deep learning model is not necessary. If there is an existing solution to a problem and that solution works, then it also may be difficult to improve on that solution with deep learning without adding new, informative sources of data. One recent example where deep learning shows no benefit over more traditional approaches is predicting the outcome of cardiac bypass surgery. Although the largest medical data set available has more than 80,000 cases, simple statistical logistic regression performs as well as more complex deep learning approaches with the categorical and numerical data available (Shahian & Lippmann, 2020). This is a common finding in many areas, and it is always important to first try simpler solutions before developing deep learning models.

6.2 Deep Learning Models Are Often Fragile and Not Quite Superhuman. Many news articles start with a statement about how AI deep networks perform better than humans on some tasks. In practice, deep networks might perform well on some limited task in the laboratory but can fail spectacularly even when inputs are varied only slightly. This section focuses on the fragility to small changes in test data.

A good example of this fragility is the sensitivity that deep learning image classifiers have to slight modifications of the ImageNet test data. ImageNet contains roughly 1.2 million training examples containing pictures from the Internet and 1000 classes. The accuracy of the top single guess from a well-trained human on ImageNet is roughly 91% (Shankar et al., 2020), and a high-performance Resnet50 model's comparable top-1 accuracy is 76.2% (Radford et al., 2021; He, Zhang, Ren, & Sun, 2016) as shown by the left-most bar in Figure 6. Human performance is not perfect because ImageNet contains unfamiliar class labels such as as the dog breeds "Leonbert," "Vizsla," and "Weimaraner" that are difficult to label correctly even with training. Other common objects like "teapot," "screw," and "padlock" are classified almost perfectly. The shaded area in the top of Figure 6 shows the range of trained human performance on ImageNet and three degraded versions of ImageNet. It can be seen that human performance remains above 90% for all degraded data sets.

The second bar from the left in Figure 6 shows how the Resnet50 system degrades when a different ImageNet test sample, called ImageNet V2, with roughly 20,000 images was created by scanning the Internet using the same procedures used to generate the original training images (Recht, Roelofs, Schmidt, & Shankar, 2019). This is a surprising dramatic reduction in performance demonstrating poor generalization on images similar to those used for training! Humans show no consistent reduction in performance on these new test data (Shankar et al., 2020).

The third bar in Figure 6 shows performance when new test data called ObjectNet include images of common household objects taken from

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

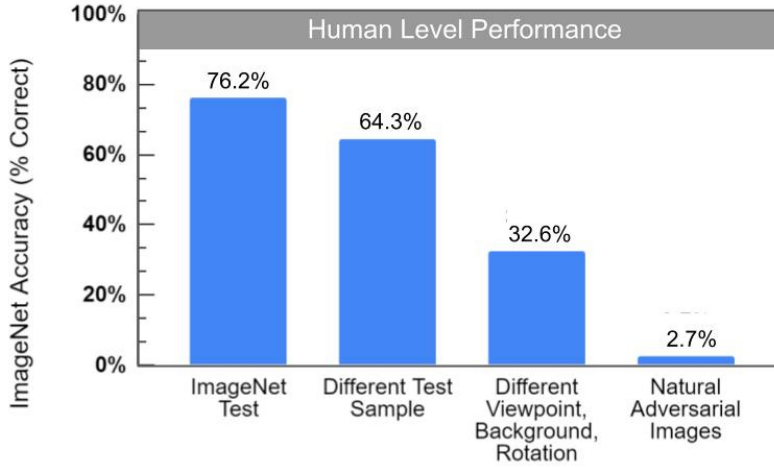

Figure 6: The four bars show ImageNet top-1 accuracy for a high-performance deep learning Resnet50 model for normal ImageNet test images and for data sets with degraded images. Human performance remains high, above 90%, even with degraded images, as shown by the upper shaded area.

different viewpoints, with objects at different rotations, and with different backgrounds (Barbu et al., 2019). Performance of Resnet50 drops precipitously to roughly 33%, while human performance remains near 90%. Finally, another study selected natural images that are difficult for ImageNet classifiers to identify but easy for humans (Hendrycks, Zhao, Basart, Steinhardt, & Song, 2021). The right-most bar in Figure 6 shows that the Resnet50 performance drops to almost zero (Radford et al., 2021) while human performance remains near 90% correct (Hendrycks et al., 2021).

When possible, fragility can be reduced by collecting more training data that better sample the variability seen during deployment. A recent study, for example, put forth the effort to collect 400 million images from the Internet using associated text as weak labels and train a classifier using 256 GPUs over two weeks (Radford et al., 2021). Performance on normal ImageNet test images was the same as shown in Figure 6, but changes in test images did not cause as large reductions in performance.

All of these studies suggest that humans use a different approach to image classification than deep learning models do. Models appear to focus more on local clues such as texture, color, local shape variations, and backgrounds that might be consistently different between ImageNet classes. We seem to have an internal model of objects and can use it to recognize objects from different viewpoints and rotations and with different backgrounds.

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

These studies suggest that deep learning is most useful in well-controlled environments where backgrounds are controlled and data in deployment are as close to training data as possible.

The same caution is true for recently developed large deep learning language models (Brown et al., 2020; Devlin, Chang, Kee, & Toutanova, 2019). They use training on large amounts of word sequences to extend inputs, answer questions, or summarize text. When given text not seen during training, they can easily be coaxed to give nonsensical false answers. This is one reason that these models are not currently in widespread use for applications such as Internet search (Metzler, Tay, Bahri, & Najork, 2021). There are numerous examples of this problem, but here is a simple one where a language model described in Brown et al. (2020) and available online (OpenAI, 2022b) answers a ridiculous question never seen before. After entering the question, "When George Washington piloted an airplane to New York, what type of biplane did he fly?" the response is, "The plane George Washington flew was a Curtiss JN-4D, also known as a Jenny." Notice how confident the answer is, providing exact details on the biplane.

6.3 Be Careful Applying Deep Learning Models in Life-Critical Situations. Fragility, overconfidence in wrong decisions, and lack of easy explanations for decisions make it problematic to use deep learning in life-critical situations. It is difficult to verify that a model will work correctly when deployed because inputs can deviate far from training data. It is also difficult to verify that the model is focusing on the relevant parts of the input and operating in a sensible manner.

There is no easy solution to these problems. When possible, simple but effective classifiers that are transparent and easy to explain can be used to assist in decision making for criminal justice, law enforcement, and medical decisions as suggested and demonstrated in Rudin (2019). It is also important that training data include rare events and unusual corner cases to make sure they are handled properly. Andrew Karpathy (2020) reveals how difficult it is even to train a simple stop-sign detector for autonomous vehicles. Multiple classifiers are required to detect occluded, hand-held, digital, school bus, and many other types of stop signs and tens of thousands of training examples are collected to train each classifier. Even with all this effort, unusual situations might arise that were not sampled in training.

Much recent research is addressing these issues. Approaches that estimate the uncertainty in deep learning model outputs are summarized in Abdar et al. (2021), and *anomaly detection* attempts to determine when inputs are too different from training data (Hendrycks, Mazeika, & Dietterich, 2019). These approaches could be used to determine when to trust a model and when to delegate a task to a human. There is also extensive recent work in the area of explainable, understandable, and fair AI (Ghassemi, Oakden-Rayner, & Beam, 2021; Linardatos, Papastefanopoulos, & Kotsiantis, 2020).

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

7 Conclusion

Deep learning models provide new tools to address difficult real-world problems. Success requires the joint cooperation of both machine learning developers and experts in an application area. It is beneficial if everyone involved in system development understands the overall development process, including the need to gather sufficient representative data for training and to carefully monitor models after deployment and retrain when necessary. Expectations can be set properly only after understanding the limits of deep learning systems, including their dependence on training data, fragility to new test inputs, lack of transparency, and a tendency to be overconfident when making predictions.

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., . . . Nahavandi, S. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, *76*, 243–297. 10.1016/j.inffus.2021.05.008

Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., . . . Katz, B. (2019). ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems*, *32*. Red Hook, NY: Curran.

Bommasani, R., Hudson, D., Adeli, E., Altman, R., Arora, S., von Arx, S., et al. (2021). *On the opportunities and risks of foundation models*. arXiv:2108.07258.

Booq, R. Y., Tawfik, E. A., Alfassam, H. A., Alfahad, A. J., & Essam, J. A. (2021). Assessment of the antibacterial efficacy of halicin against pathogenic bacteria. *Antibiotics*, *10*(12). https://www.mdpi.com/2079-6382/10/12/1480. 10.3390/antibiotics10121480, PubMed: 34943692

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., . . . Amodi, D. (2020). *Language models are few-shot learners*. arXiv:2005.14165.

Chen, J., J., & Ran, X. (2019). Deep learning with edge computing: A review. In *Proceedings of the IEEE*, *107*(8), 1655–1674. 10.1109/JPROC.2019.2921977

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., . . . Wei, F. (2021). *WavLM: Large-scale self-supervised pre-training for full stack speech processing*. arXiv:2110.13900.

Chen, X., Xie, S., & He, K. (2021). *An empirical study of training self-supervised vision transformers*. arXiv:2104.02057.

Creswell, A., White, T., Dumolin, V., Arulkunaran, K., Sengupta, B., & Bharath, A. A. (2017). *Generative adversarial networks: An overview*. arXiv:1710.07035.

Dai, Z., Liu, H., Le, V., & Tan, M. (2021). *CoAtNet: Marrying convolution and attention for all data sizes*. arXiv:2106.04803.

Decario, N., & Etzioni, Z. (2021). *America needs AI literacy now*. Seattle, WA: Allen Institute for AI. https://pnw.ai/article/america-needs-ai-literacy-now/72515409

Devlin, J., Chang, M.-W., Kee, K., & Toutanova, K. (2019). *BERT: Pre-training of deep bidirectional transformers for language understanding*. arXiv:1810.04805.

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

Fridman, L. (2020). *Deep learning state of the art 2020*. https://lexfridman.com/files/slides/2020_01_06_deep_learning_state_of_the_art.pdf

Ghassemi, M., Oakden-Rayner, L., & Beam, A. (2021). The false hope of current approaches to explainable artificial intelligence in health care. *Lancet Digital Health*, *3*(11), e745–e750. 10.1016/S2589-7500(21)00208-9, PubMed: 34711379

Goodfellow, I., Pouget-Avadie, J., Mirza, M., Xu, Bing, Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems*, *27*. Red Hook, NY: Curran.

Google. (2022). *Teachable machine: Train a computer to recognize your own images, sounds, and poses*. https:/teachablemachin.withgoogle.com/

Greenspan, H., van Ginneken, B., & Summers, R. M. (2016). Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, *35*(5), 1153–1159. 10.1109/TMI.2016.2553401

Gulshan, V., Peng, L., & Coram, M. (2016). Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA, 316*(22), 2402. 10.1001/jama.2016.17216, PubMed: 27898976

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770–778). Piscataway, NJ: IEEE.

Hendrycks, D., Mazeika, M., & Dietterich, T. (2019). *Deep anomaly detection with outlier exposure*. arXiv1812.04606.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., & Song, D. (2021). Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 15262–15271). Piscataway, NJ: IEEE.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. 10.1162/neco.1997.9.8.1735, PubMed: 9377276

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, *11*(1), 63–90. 10.1023/A:1022631118932

Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., . . . Adam, H. (2019). Searching for MobileNetV3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 1314–1324). Piscataway, NJ: IEEE.

Ignatov, A., Timofte, R., Chou, W., Wang, K., Wu, M., Hartley, L., & Vab Gool, L. (2018). AI benchmark: Running deep neural networks on android smartphones. In *Proceedings of the European Conference on Computer Vision Workshops*. Berlin: Springer.

Karpathy, A. (2020). *AI for full-self driving at Tesla*. Paper presented at the 5th Annual Scaled Machine Learning Conference. https://watch?v=hx7BXih7zx8

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8110–8119). Pisacataway, NJ: IEEE.

Kissinger, H., Schmidt, E., & Huttenlocher, D. (2021). *The age of AI and our human future*. New York: Little, Brown.

Koenecke, A., Nam, A., Lake, E., & Goel, S. (2020). Racial disparities in automated speech recognition. In *Proceedings of the National Academy of Sciences*, *117*(14), 7684–7689. 10.1073/pnas.1915768117

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. 10.1038/nature14539, PubMed: 26017442

Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2020). Explainable AI: A review of machine learning interpretability methods. *Entropy*, *23*(1), 18. 10.3390/e23010018, PubMed: 33375658

Lippmann, R. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine*, *4*(2), 4–22. 10.1109/MASSP.1987.1165576

Metzler, D., Tay, Y., Bahri, D., & Najork, M. (2021). Rethinking search: Making domain experts out of dilettantes. *ACR SIGIR Forum*, *55*, 1–27. 10.1145/3476415.3476428

Microsoft. (2022). *Lob: Machine learning made easy*. https://www.lobe.ai/

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., & Carter, S. (2020). An overview of early vision in InceptionV1. *Distill*. https://distill.pub/2020/circuits/early-vision.

OpenAI. (2022a). *OpenAI Microscope*. https://microscope.openai.com/models

OpenAI. (2022b). *OpenAI Playground: Text completion task*. https://beta.openai.com/playground

Radford, A., Kim, J., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., . . . Sutskever, I. (2021). *Learning transferable visual models from natural language supervision*. arXiv:2103.00020.

Rajpurkar, P., Irvin, J., Zhu, K., Yang, B., Mehta, H., Duan, T., . . . Ng, Y. (2017). *CheXNet: Radiologist-level pneumonia detection on chest x-rays with deep learning*. arXiv:1711.05225.

Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do ImageNet classifiers generalize to ImageNet? In *Proceedings of the 36th International Conference on Machine Learning*. https://proceedings.mlr.press/v97/recht19a.html

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, *1*(5), 206–215. 10.1038/s42256-019-0048-x

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., . . . Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211–252. 10.1007/s11263-015-0816-y

Shahian, M., & Lippmann, R. P. (2020). Commentary: Machine learning and cardiac surgery risk prediction. *Journal of Thoracic and Cardiovascular Surgery*. 10.1016/j.jtcvs.2020.08.058

Shankar, V., Roelofs, R., Mania, H., Fang, A., Recht, B., & Schmidt, L. (2020). Evaluating machine accuracy on ImageNet. In *Proceedings of the International Conference on Machine Learning* (pp. 8634–8644).

Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N., . . . Collins, J. (2020). A deep learning approach to antibiotic discovery. *Cell*, *180*(4), 688–702. 10.1016/j.cell.2020.01.021

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking model scaling for convolutional neural networks*. arXiv:1905.11946.

Vanschoren, J., & Yeung, S. (2021). *NeurIPS Datasets and Benchmarks Track*. NeurIPS 2021 Conference. https://datasets-benchmarks-proceedings.neurips.cc/paper/2021

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025

Vaswani, A., Shazeer, N., Parmer, N., Uszkoreit, J., Jones, L., Gomez, A., . . . Polosukhin. (2017). Attention is all you need. In I. Guyon, Y. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems*, *30* (pp. 5998–6008). Red Hook, NY: Curran.

Vinuesa, R., Azizpour, H., Leite, I., Balaam, M., Dignum, V., Domisch, S., . . . Nefini, F. F. (2020). The role of artificial intelligence in achieving the sustainable development goals. *Nature Communications*, *11*(1), 233. 10.1038/s41467-019-14108-y

Wang, X., Han, Y., Leung, V., Niyato, Yan, X., & Chen, X. (2020). Convergence of edge computing and deep learning: A comprehensive survey. *IEEE Communications Surveys Tutorials*, *22*(2), 869–904. 10.1109/COMST.2020.2970550

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, . . . Barzilay, R. (2019). Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, *59*(8), 3370–3388. 10.1021/acs.jcim.9b00237, PubMed: 31361484

Received February 4, 2022; accepted April 4, 2022.

Downloaded from http://direct.mit.edu/neco/article-pdf/35/3/287/2071846/neco_a_01518.pdf by guest on 18 March 2025