Maximum Blue Green

Deep Learning Overview

By Sophie Saunders, ECE '20

Introduction

Although you may not have heard of it until recently, machine learning (ML) actually originated in the 1950s. The famous first application was in IBM's 1956 checker-playing program. By definition, ML is when a computer is able to learn without explicit programming. In the checkers example, IBM's program didn't identify every possible combination of checkers on the board and what the next move would be. Instead, by understanding the game's objective and how to get closer to winning, the computer could make its own decisions about the optimal next move. ML is a subset of artificial intelligence (AI), which is a more general term for when computers mimic human behavior. This report focuses on one type of ML that has become especially popular over the last few decades, known as deep learning.

Self-taught Computers

Artificial neural networks

Deep learning, like its relatives discussed above, mimics human behavior and learns without explicit programming, but it is unique because it learns through *artificial neural networks*. By hugely over-simplifying, we can describe the human brain as the connection of 100 billion nerve cells called neurons. In order for an axon (part of the neuron) to fire a signal, the signal must surpass some threshold. Deep learning imitates neurons using *perceptrons*, which are vectors of input values with associated weights. With input values x, y and weights A, B, the combined value is Ax+By. The output of the perceptron is 0 if Ax+By is less than some assigned threshold, or 1 if it surpasses the threshold. More complicated models also exist that modernize the perceptron, such as the sigmoid neuron which can output a range of values from zero to one (which indicates the probability of that outcome).

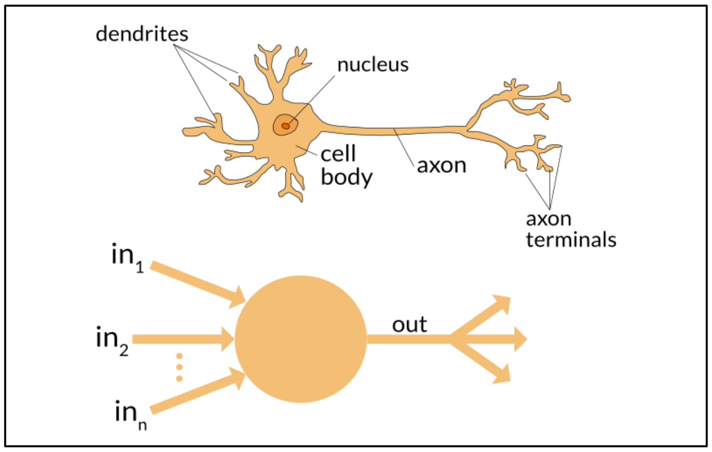

Figure 1. Comparison of neuron and perceptron

Mapping inputs to outputs

A deep learning model has three layers: input, hidden, and output. Inputs are accepted in numerical form (whether images, sounds, accelerations, heights, or more) and must be labeled with the corresponding output value. The hidden layer is the perceptron algorithm that maps inputs to the outputs. The user does not need to see or understand this algorithm in order to use the deep learning model. For example, the user could provide a set of labeled images as inputs that depict either a dog or a cat. The model would extract features from the image (possibly colors, overall shapes, line segments, etc.) and give them weights, creating some algorithm to differentiate the animals. By creating an algorithm from known, labeled images, we can then classify new, unlabeled images as dogs or cats.

Quantifying Success

Minimizing loss

To begin training the neural network, random values are assigned as weights, which are then varied systematically to get the best results. The "best" weights are the ones that minimize the loss, or the error, in the model. One iterative approach is referred to as the *gradient descent optimization algorithm*, which repeatedly calculates the loss associated with those set weights and updates the parameters accordingly. How that will look depends on our learning rate – how drastically the model parameters are changed each iteration.

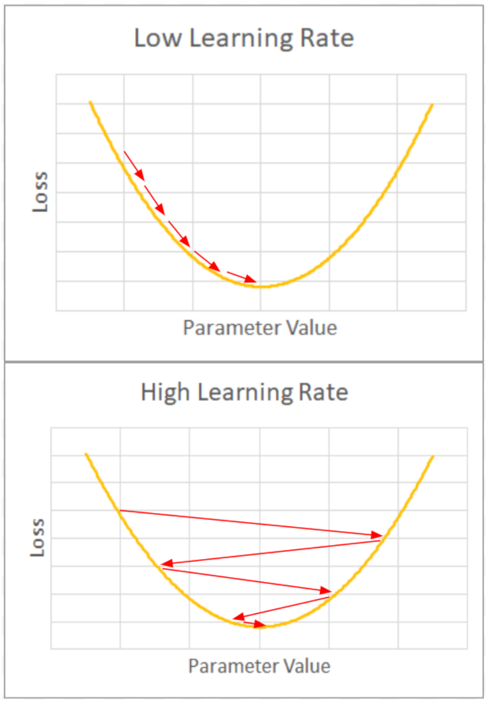

Figure 2. Finding minima by varying parameter value with low and high learning rates

Because deep learning can be used to generate many different types of outputs, different types of loss functions are required. For example, deep learning can be used for classification (ie, is that a picture of a cat or a dog?) or for regression (ie, how many dogs are in that picture?). Cross-entropy measures loss for and mean squared error measures loss for regression.

Preventing under/overfitting

Another way of understanding optimization is that a good model must neither underfit nor overfit. Underfitting means that the model does not accurately depict the sample data -- for example, a straight line may model a curve. Overfitting means that the model too precisely describes the exact sample data, rather than the general trend. The model may zig-zag to exactly cross through all the points rather than allowing for some error. A common-sense example of underfitting would be a model that says any image containing a four-legged animal shows a dog. This model does not accurately distinguish images with dogs from images without, because many animals have four legs. On the other hand, an overfitting model might determine that in all images with dogs, there must be blue sky in the background and a white four-legged animal with a tail and red collar. This definition, however, is too specific to the training images provided and would not correctly identify any image of a dog. We want to train the model only until we arrive at some middle point between under and overfitting, which is the sweet spot. Sure enough, this spot is also where loss is at a minimum.

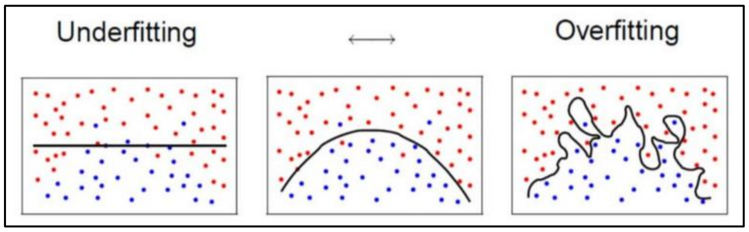

Figure 2. Underfitting, ideally fitting, and overfitting to distinguish red from blue dots

Deep Learning Tools

Popular libraries for deep learning include Tensorflow, Keras, PyTorch, Caffe, and many more. They range in terms of simplicity, configurability, and documentation. A user might select a specific library based on their experience level, the complexity of their project, and their desired programming language. Interestingly enough, almost all deep learning software is designed for use in the Python programming language (although MATLAB and C/C++ also have their own). Beyond the fact that Python has generally spiked in popularity over the last decade, there are a few reasons why it is such an attractive choice for deep learning.

Python is well-known for being simple, readable, intuitive, and concise. This allows the programmer to focus on the challenge presented by the ML model rather than the challenge of using a difficult programming language. It is used across fields and industries and is also platform independent such that the same code that runs successfully on a Mac will also work on a Windows or Linux machine. Python is well-documented and has a huge community online to help beginners and professionals alike.

Big tech companies have seen the direction that technology is heading and created deep learning software like Apple's CoreML, Google's Tensorflow, Microsoft's CNTK, and Amazon's AML. None of these require a mathematical understanding of neural networks to set up or utilize. Deep learning has evolved over the past seventy years to become more user-friendly and publicly accessible.

Conclusion

Although ML seems intimidating – and deep learning even more so – the resources available make it understandable at any level of computer-savviness. In a team without any prior ML knowledge, we were able to create a deep learning model that performs binary classification of accelerometer data, labeling it as either a squat or a deadlift. We chose Keras (which is a user-friendly library that runs on top of Python's Tensorflow library) as our deep learning software and followed online tutorials to create a multi-layer model that can identify exercises much better than a human. Deep learning has limitless future potential and we can't wait to see what problems it solves next.

References

1. Brownlee, Jason. "Loss and Loss Functions for Training Deep Learning Neural Networks." *Machine Learning Mastery*, 22 Oct. 2019, machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/.

2. "How Does the Brain Work?" *Ncbi.nlm.nih.gov*, U.S. National Library of Medicine, 31 Oct. 2018, www.ncbi.nlm.nih.gov/books/NBK279302/.

3. "The IBM 700 Series." *IBM100 - The IBM 700 Series*, www.ibm.com/ibm/history/ibm100/us/en/icons/ibm700series/impacts/.

4. "The Perceptron." *Neural Networks - Neuron*, cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/Neuron/index.html.

5. "Reducing Loss: Machine Learning Crash Course." *Google*, Google, developers.google.com/machine-learning/crash-course/reducing-loss/video-lecture.

6. "What Is Deep Learning?" *SAS*, sas.com/en_us/ insights /analytics/deep-learning.html.