Forecasting & Futurism

July 2014 – Issue 9

An Introduction to Deep Learning

By Jeff Heaton

Deep learning is a topic that has seen considerable media attention over the last few years. Many large technology companies have invested heavily in deep learning. In January of 2014, Google purchased DeepMind (a deep learning startup) for $400 million. Deep learning is being applied to the fields of robotics, computer vision, and natural language processing. Deep learning is successful because it learns by a hierarchical system of features that bears similarity to the human mind. Deep learning also works well with modern technologies such as grid computing and General Purpose Graphics Processing Units (GPGPU).

Deep learning does hold great potential for data science. However, deep learning works somewhat differently than many of the more familiar statistical models. In this article I will introduce deep learning and show how it relates to other techniques in the field of data science. I will also show how deep learning has application to the type of unstructured data seen by the insurance industry.

TOWARD COMPOSITE MODELS

Initially, you may want to compare deep learning to statistical models and machine learning models such as neural networks, support vector machines, linear regression, generalized linear models (GLM) and others. It is very important to remember that deep learning is not a specific model. Rather, deep learning is a means of combining several models together to form a composite model. The individual components will retain autonomy and can be trained independently.

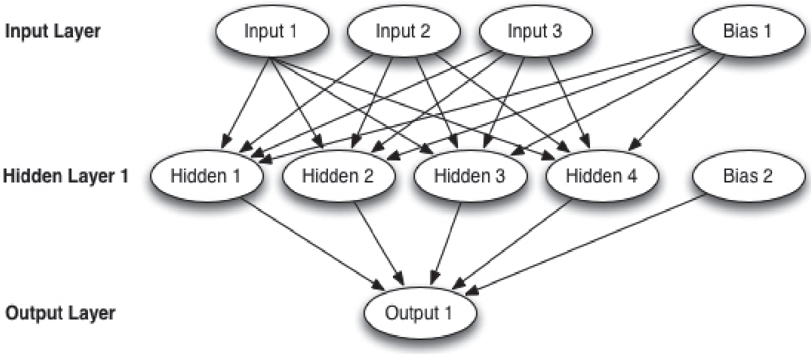
Over the last five years, boosting and ensemble learning have become two very popular techniques for producing composite machine learning models. Neither boosting, nor ensemble learning, specifies exactly what models make up the resulting composite model. The primary high level difference between boosting and ensemble learning is that boosting uses a homogeneous set of models, whereas an ensemble is heterogeneous. An ensemble is much like an orchestra producing one song with many different instruments.

Like boosting and ensemble learning, deep learning also produces a composite model. However, the deep learning model offers some unique features that are not seen in other machine learning models. Deep learning allows individual parts of the model to be trained independently of the others. Deep learning is typically applied to neural networks. However, this is by no means a necessity. Yichuan Tang, of the University of Toronto, introduced the use of deep learning for support vector machines.[1]

DEEP LEARNING ARCHITECTURE

Consider the typical multi-layer perceptron (MLP), or neural network. Such a network has an input layer, zero or more hidden layers, and an output layer. Most neural networks contain one single hidden layer. Figure 1 shows just such a network.

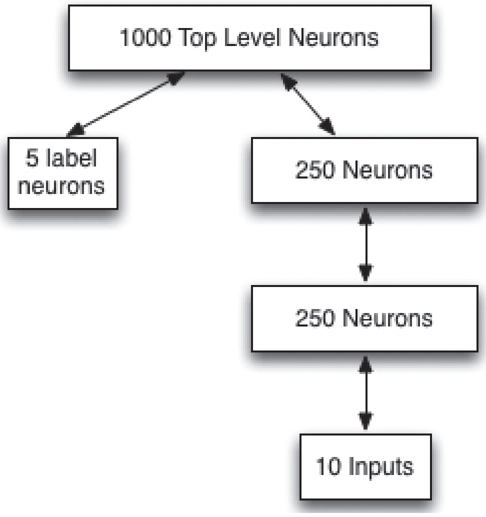
Figure 1: Shallow Multi-Layer Perceptron (MLP, Neural Network)



The above diagram shows the inputs, hidden layers, outputs and bias neurons. Weights connect these neurons together. Weights control the sigmoidal curve of the neuron's output. Bias neurons allow the neuron's sigmoidal output curve to be shifted left or right in the x direction. Most neural networks are shallow, and have a single hidden layer. However, it is possible to create neural networks with two or more hidden layers, as seen in Figure 2.

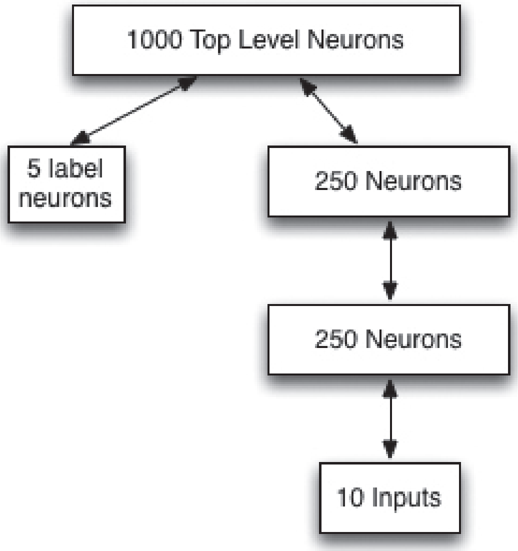
CONTINUED ON **PAGE 18**

Figure 2: A Deeper MLP (3 hidden layers)



The above neural network contains a total of three hidden layers. Most research indicates that more than a single hidden layer is counterproductive.[2] Furthermore, additional hidden layers greatly lengthen the training time for the neural network. Is a deep belief network simply a neural network that has a large number of hidden layers? Yes and no.

Figure 3: shows an overview of deep learning architecture.



Deep learning recognizes that you may not always have labels for all of the data you have collected. Deep learning allows the network to be trained using both supervised, and unsupervised techniques. You might not know the desired outcome for every item in your data set. This is OK with deep learning.

This approach very much models the way the human brain functions. A human child sees many different types of vehicles before they ever learn the difference between a car and motorcycle. However, years of learning taught that human to identify what features a vehicle has. Features describe how many tires a vehicle has, its shape, color and size. All of these features are rolled up into the person's final classification of what sort of vehicle this is.

Simultaneous supervised and unsupervised training is what lets a DBNN get away with being "deep." It is not practical to train a traditional neural network in both a supervised and unsupervised manner at the same time. The vanishing gradient problem causes the backpropagation derivatives to shrink as each new layer is added. Additionally backpropagating through many layers is computationally expensive.

Training a DBNN is usually accomplished by the following steps (shown in Figure 3).

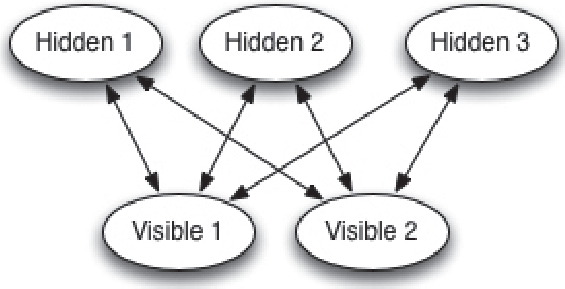
1. Train the first layer (250 neurons) with the 10-input data provided in an unsupervised way.

2. The first layer has now learned a representation of the data that is used to train the second layer in an unsupervised way.

3. This process continues until we have trained the top 1,000 neuron layer.

4. Finally, we use our labels to train a logistic regression (or similar model) based on the features extracted from the top 1,000 neuron layer. Labels identify what we are ultimately trying to predict with our model and data. For example, in an underwriting system, labels might be the final underwriting decision.

Figure 4: Restrictive Boltzmann Machine (RBM)



The key to this process is that we are hierarchically learning features first from the training data, and then features built upon features from the lower levels. We are not training the entire model at once. Each layer is trained independent of the others. Finally, the labels that we have are used to perform a more traditional gradient-based fine tuning of the final output of the model. This training method is what truly separates a DBNN from a regular neural network with a large number of hidden layers.

RESTRICTIVE BOLTZMANN MACHINES

Deep learning does not imply what makes up each level of the model. However, DBNN's are usually made up of Restrictive Boltzmann Machines (RBM). An RBM is essentially a simple neural network made up of visible and hidden elements. A sample RBM is shown in Figure 4.

The RMB is said to be restricted, because connections only occur between visible and hidden nodes. Some variants of RBM do allow lateral connections among visible nodes. However, no RBM model allows connections among the hidden nodes.

A full discussion of RBM's is beyond the scope of this article. However, one of the most challenging aspects of an RBM is that all input and output is binary. You cannot directly use continuous numbers with an RBM. One of the biggest challenges, for using an RBM, is to construct your input data as a binary vector. For computer vision problems, the input is often a pixel map. For non-graphical data, you need to get a little more creative.

DEEP LEARNING AND UNSTRUCTURED DATA

Unstructured data is a very active, and challenging, area of data science research. There are many different ways to handle unstructured data. A common task in unstructured data is to classify documents. You might want to cluster similar documents, or you might want to find similar documents given a starting example document. Most statistical models, DBNN's included, require the input data to be represented as a numeric vector.

There are many different ways to represent a document as a numeric vector. One of the most common is the "Bag of Words" algorithm. For example, to create a 2,000 element vector, the "Bag of Words" algorithm proceeds as follows.

1. Remove all "stop words" (i.e., "the," "and," "or," etc.) from the document.
2. Remove all punctuation from the document.
3. Change all words to a common stem (e.g., "people" becomes "person").
4. Perform a frequency count of all remaining words.
5. Arrange the counts of the top 2,000 alphabetically (or any consistent ordering). This is your input vector.

Because input vectors must be consistent you must always choose the same 2,000 words over all documents that you will classify. For example, if you were classifying Wikipedia articles you would build your 2,000 word vector of the most common "non-stop words" in Wikipedia. Unfortunately, this word frequency vector is not binary, as required by a DBNN. To convert the frequency vector to binary you typically establish a threshold count. Any word that has a frequency above this count is represented by 1, otherwise 0.

Attending Physician Statements (APS) are a common form of unstructured data seen in the insurance industry. Using machine learning models to classify and compare APS statements could be very useful to the life insurance industry.

CONTINUED ON **PAGE 20**

AN INTRODUCTION TO DEEP LEARNING | FROM PAGE 19

I am currently researching the applicability of deep learning, as well as other machine learning algorithms, to APS analysis.

GETTING STARTED WITH DEEP LEARNING

One of the best sources of information for deep learning is the site *http://www.deeplearning.net.* This site is maintained by some of the most active researchers in the field of deep learning. This site includes a very helpful tutorial at the following URL.

http://www.deeplearning.net/tutorial/

The Python programming language is a very popular choice for deep learning research. All of the examples contained at the above URL are written in Python. They also make use of the Theano Mathematical package for Python. Theano is described as a CPU (central processing unit) and GPU (graphics processing unit) math expression compiler. Theano handles the mathematical processing behind deep learning.[3] Theano is capable of using a higher-end GPU to speed up computations by up to 140 times. GPU's in the $500 USD range can typically achieve this level of performance.

The above tutorials start with familiar statistical models, such as logistic regression. New techniques and models are then added as the tutorial progresses eventually to deep learning. ▼

Jeff Heaton

Jeff Heaton is EHR data scientist at RGA Reinsurance Company and author of several books on artificial intelligence. He can be reached at *jheaton@rgare.com.*

ENDNOTES

[1] Deep Learning using Linear Support Vector Machines, http://deeplearning.net/wp-content/uploads/2013/03/dlsvm.pdf

[2] How many hidden layers should I use? http://www.faqs.org/faqs/ai-faq/neural-nets/part3/

[3] Theano, http://deeplearning.net/software/theano/