# Intro to Deep Learning by Sandro Skansi

**by Rick Rejeleene**

This is the second, Deep Learning book that I am reading as my second step entering, Artificial Intelligence.

The first was by Eugene Charniak. I would recommend you to read it. I think it was a gentle introduction to the entire field of Machine Learning. To Climb a mountain, we need to make a small step, I believe being gentle is good at first.

Deep Learning is sub-set of Machine Learning that specifically uses Artificial Neural Networks.

Although I do not completely understand the mathematical vocabulary to express elegant solutions, I think, this is a fair attempt for me to understand it.

I am content with my approach and believe that in the next few iterative attempts, I will be able to articulate it better.

I write this to help a lay-reader to get a glimpse and as a first step to open under the hood understanding, for popular buzz-word Artificial Intelligence. I am drawn towards the field due to my love of Philosophy.

**Outline:**

**Chapter 1: Logic to Cognitive Science.**

My own thoughts on this chapter is that, it is important to understand historical context to get a scoop of how beautiful, the field of Machine Learning arose to the modern scene of academics and industry.

The key thoughts in this chapter are Good Old Fashion A.I, which concerns itself symbolic logic, mathematical precision against connectionism, that has cognitive science, psychology and philosophical predicates.

Using the former, Researchers were building expert systems and the later, we know that it is taking over modern systems in image recognition, computer vision, automated systems.

Sandro Skansi says, Artificial Intelligence has its root under Leibniz. Leibniz is one of my favorite philosophers in history. Leibniz concealed two concepts, characteristica universalis and calculus ratiocinator. Characteristica universalis is an idealized language that can translate all science and any principles. calculus ratiocinator is the machine, whether software or hardware, you decide.

Deep Learning is the connectionist tribe using artificial neural network that has strongholds in some aspects of Artificial Intelligence.

This Chapter gives you a glimpse of how interdisciplinary the field of Deep Learning has been, from psychiatry, philosophy, mathematics and biology. AI research concerns mostly with understanding, modeling and formalizing the interactions of cognitive processes.

Sandro Skansi says Artificial Intelligence is ***Philosophical Engineering.***

Why does he say this? Because, Artificial intelligence is trying to replicate intelligence.

To summarize the chapter, ACM classification has the following formal classification for Artificial Intelligence:

*-General Learning, Adaptive Systems*
*-Pattern Recognition and Speech Recognition*
*-Theorem Proving, Problem Solving, Logic*
*-Knowledge Representation, Language and Software Systems*
*-Reasoning under uncertainty,*
*-Robotics, Agent technology, Machine vision*

**Chapter 2: Mathematical and Computational Pre-Requisite:**

I will try my best attempt to understand intuitiveness of concepts rather bore you with mathematical formalization. I believe mathematical formalization is important to learn to play and practice A.I. It is similar to a musician learning to play various types of music through understanding musical notations. I believe in my fourth or fifth iteration, I will bring you the magic of mathematical formalization.

I'm going to concise the important topics that are covered in this chapter.

**The Main Engine of Deep Learning is Backpropagation.**

Backpropagation consists of gradient descent and to move along a gradient. The Gradient is vector of derivations.

There's many examples of vectors, matrices, linear programming, concepts from statistical and probability theory in this Chapter. Finally, there's also examples of Logic and Turing machines. Logic is foundation of mathematics, turning machine is the basic theoretical concept that a computing machine is built upon. If you come across Aristotle and until George Boole, you can get a fair understanding and evolution of it.

Skansi goes into a brief intro to Python Programming. This might be useful for non-CS majors.

**Chapter 3: Machine Learning Basics:**

This Chapter is going to give basics in the field of Machine Learning.

Machine Learning is the subfield of artificial intelligence and cognitive science.

There's three branches within Machine Learning: supervised, unsupervised and reinforcement learning. Deep Learning is a special approach in Machine Learning that covers all branches and seeks to extend to other areas of Artificial Intelligence.

This Chapter has the following material:

a) ***Elementary Classification:***
b) ***Naive Bayesian Classifier:***
c) ***A Simple Neural Network: Logistic Regression:***
d) ***MNIST Dataset***
e) ***Learning without Labels: K-Means***
***f) Learning Language: bag of word representation***

***a) Elementary Classification:***

Supervised learning is elementary classification. Consider the problem in an image classification, “is car” or “not car.” I ask the reader to get the book for understanding the details.

Evaluating Classification Results involve using confusion matrix, which has concepts of true positive, true negative, false positive, false negative.

Categorical features are very common. Machine Learning algorithms cannot accept categorical features, eg: we have a column named Dog. We allow only binary values in the column, and convert it into 1 or 0. This is called one-hot encoding.

***b) Naive Bayesian Classifier:***

This uses basic bayesian classifier, the core assumption is that variables are independent. Therefore, it cannot handle dependencies in features for classifier.

***c) A Simple Neural Network: Logistic Regression:***

This is an important concept that is going to help build our understanding of neural networks and deep learning.

Within Supervised learning, we divide it into two types, one is to predict a class, the next is predicting a value. For the first, we use naive bayesian classifier. For the second, predicting a value, we explore logistic regression. Logistic Regression is “not a regression algorithm but a classification algorithm.” Machine Learning community uses it as a classifier instead of regression model.

Logistic function is the main component of a logistic regression. If we think logistic regression as simple neural network, we are not committed to logistic function — in this, it is non-linearity, a component which enables complex behavior.

***d) MNIST Dataset:***

MNIST Dataset is modification of national institute of standard consisting of handwritten digits. Geoffrey Hinton called MNIST the fruit fly of machine learning because a lot of research is performed on it.

### *e) Learning without Labels: K-Means*

For two algorithms for unsupervised learning, there's two algorithms here: PCA and K-means.

PCA represents a branch of unsupervised learning called distributed representations. It is one of the most important topics in deep learning. PCA is the most simple algorithm for building distributed representations.

Another algorithm called clustering, is similar to PCA. It captures similarity in n-dimensional space.

K-means is un supervised learning, which means there's no labels or targets in the dataset. It produces clusters of data.

Correlation is when two data-points move together linearly or has similarity. For eg: People over 180 cm are more likely to be above 80 kilos. If two features are highly correlated, then it is hard to tell them apart.

Ideally we want to find features which are not correlated that captures underlying component. In Statistics, it is called latent variables. On doing this, we are able to represent something called distributed representations.

Building distributed representation is the essence of what artificial neural networks do. Every layer builds own distributed representation and this facilitates learning.

### *f) Learning Language: bag of word representation*

We have seen numerical features, ordinal features and categorical features. We explore Natural Language Processing. In it, it let's dive into processing language by using a simple model, the bag of words.

In Natural Language Processing, we use terms like a corpus, fragments. A corpus is whole collection of texts we have, they can be composed into fragments. A fragment can be single sentence, paragraph or multi page documents. Fragment is what we use for training sample. Eg: A PhD Thesis is one fragment, if we analyze sentiment in social media, each comment is one fragment.

**Chapter 4: FeedForward Neural Networks:**

As we saw in previous chapter, back propagation is the core method of learning in Deep Learning. I think, I did not explain what is Deep Learning in previous chapters. It is machine learning with deep artificial neural networks. In this chapter, we explore simple neural networks called feedforward neural networks.

A neural Network is made of simple basic elements in logistic regression. Logistic regression is the most simple neural network.

Every element that holds an input is called neuron. The logistic regression, then has a single point where all inputs are directed and this is its output. The difference in artificial neural network is that there is a hidden layer between input and output layer.

Every neuron is connected to all neurons of next layer, it might be multiplied by so called weight. The flow of information goes from first layer, second layer and third layer neurons. Different layers have different non-linearities.

We represent them in vectors and matrices.

***Perceptron Rule:***

Skanski says learning in neurons is updating of weights and biases during back propagation. An early procedure for artificial neurons is perceptron learning. It consists of binary threshold neuron (binary threshold units). Perceptron learning rule looks like a modificed logistic regression.

An example of Perceptron training would look like the following:
- Choose training set
- Predicted output matches output label, do nothing
- If Perceptron prediction is 0, but if it should have predicted a 1, then add input vector to weight vector
- If Perceptron prediction is 1, but if it should have predicted a 0, then subtract input vector from weight vector

***Delta Rule:***

The main issue in making multi layer perceptron is that it doesn't know how to expend perceptron learning to work with multiple laters.

As it doesn't know, the option seems to abandon but we mentioned a rule called back propagation. To explain this, Skansi gives an example of predicting price of a meal over repeated buys.

We want to find true price of a meal across a time using weights. In this, the learning rate controls how much error is handed when individual weights are to be updated. It is capable of learning weights across layers.

***From Logistic Neuron to Back propagation:***

Delta rule works for a simple neuron called linear neuron. To make the delta rule work, we would need a function that measures if we got the result right, if not, by how much did we miss. It is called error function or cost function.

We use derivatives to learn weights of logistic neuron.

**Back propagation:**

Back propagation is using derivatives to learn weights of logistic neuron, except that it is applying more than once to back propagate the errors through layers.

Back propagation of error is basically gradient descent. Gradient descent is optimization algorithm for finding minimum of a function. But we bump into issues: it takes too long time, second by changing weights, we will not find out whether a combination of them would work better.

While first of these problems can be overcome by gradient descent, the second is only partially resolved. It is usually called local optima. The third problem is near the end of learning, the changes will be too small. Back propagation also has this problem, and it is usually solved by using dynamic learning rate which gets smaller through learning progresses. We get a method called, finite difference approximation.

In the hidden layer of neural network, we have randomly initialized weights and raises, multiply them with inputs, add them together and take them through logistic regression which flattens them to a value between 0 and 1. We do it one more time. At the end, we get a value between 0 and 1 from logistic neuron in output layer. Everything below 0.5 is 1 0 and above is 1.

In doing many passes, we find that error has decreased. Next is an example of Python script for shopping basket classification of users abandoning it.

**Chapter 5: Modifications and Extensions to a Feed-Forward Neural Network:**

If we have a classifier in 2D space, the classifier draws a very straight line, then we have one with high bias. This can generalize well, the classifier for new points (test error) will be similar to old points (training error). This is called under fitting. On the other hand, if the classifier draws an intricate line to include every X and none of Os (Consider X and O as classes), then we have high variance ( low bias) which is called overfitting.

Emprically we want to land somewhere between over fitting and under fitting. The approach to take to find this is called regularization.

Regularization means adding a term to error function. There's two most common type of regularization: L1 and L2 regularization.
L2 is known as weight decay, ridge regression, tikhonov regularization. During learning procedure smaller weights will be preferred in L2 regularization. It tries to push down square of weights, whereas L2 is concerned with absolute values which is linear, therefore L2 will penalize large weights. L1 regularization makes more weights slightly smaller.

***Learning Rate:***

Learning rate is an example of hyper parameter. Every neural network is a function that assigns a given input and a class label (output). The way it solves this is through operations and parameters given. Operations involve logistic function, matrix multiplication.

A hyper parameter is any number that is used in a neural network which cannot be learned by the network.

***Stochastic Gradient Descent:***

Let's see how back propagation works
- We take a training sample at a time
- We pass it through the network
- We record the squared error for each
- We calculate mean squared error
- Once we have mean squared error, we back propagate it using gradient descent to find better set of weights.
- After this, we have finished one epoch of training.

**Chapter 6: Convolutional Neural Networks:**

Convolutional Neural Networks were invented by Yann LeCun. It was built by ideas of David Hubel and Torstein presented in 1968, which won them Nobel prize in Physiology and Medicine.

They discovered receptive field, which is used to describe between parts of visual fields and individual neurons which processes the information.

The step to building convolutional network is flattening images (2D arrays) to vectors. The second step is one that will take image vector to single workhorse neural which will be in charge of processing. We use logistic regression for this, but we use a different activation function, although structure remains the same.

A convolutional neural network is a neural network that has one or more convolutional layers.

Skanski then gives a full example of building a convolutional neural network using Keras in Python. On using it, we can classify text as it is setup for pattern recognition in images.

**Chapter 7: Recurrent Neural Networks:**

Feedforward Neural Networks can process vectors, convolutional neural network can process matrices (translated into vectors). But how do we process sequences of unequal length? This can be seen as a problem of learning sequences of unequal length.

Audio processing is an example of how we might do this. We need a different architecture for this. We use feedback loops that feed output back into layers as inputs. This is called recurrent neural networks. The most important recurrent neural networks are long short term memory networks or LTSMs invented by Hochreiter and Schmiduber.

In Recurrent Neural Networks, we do not simply add new layers but by adding recurrent connections on hidden layers.

***Elman Networks:***

This is the most simple recurrent network called Jordan network. It is not used in industry but used as a stepping stone to understand complicated LSTM. I ask the reader to dive into the book to understand architecture of LTSMs. Skanski goes into using recurrent neural network for predicting words.

**Chapter 8: Auto-Encoders**

An Auto Encoder is three layered feed forward neural network. It is a form of unsupervised learning. Skanski explains the mathematical intuition, formalization of auto encoders.

If we think of auto encoders as lego bricks, then it can be stacked together. The result of auto encoder is not output layer but activation in middle layer. Skanski gives an example through Keras in Python. Next, he goes in depth about the famous, “Cat paper”

This paper is written by researchers from Google, they ask the question — can we use neural networks capable of learning to recognize cats by just watching YouTube Videos?

**Chapter 9: Neural Language Models:**

A Neural Language model is distributed representation of words and sentences. They are learnt representations, which means they are numerical vectors. Word embeddings are numerical representation of word or words. A famous example of neural language model is Word2Vec Model, which learns vectors to represent words with simple neural network.

It can be built using two different architectures, skip gram and Word2Vec. We use a shallow feedforward neural network in which input layers receive word index vectors. Skanski goes into using Word2Vec in Code through Keras in Python.

**Chapter 10: Overview of architectures:**

We are entering the final two chapters, in this, Skanski gives an outline of frequently used neural network architectures.

Energy based models are a class of neural networks, the simplest is hopfield network. It is made up of neurons and all these neurons are connected among weights.

Next we turn to Boltzmann machines and Restricted Boltzmann machines. These are another type of architecture in neural networks. Boltzmann machine is a type of stochastic recurrent neural network. They are used for learning, internal representations. Restricted Boltzmann Machines are just Boltzmann machines where there is no connection between neurons of same layer. They have a visible layer and a hidden layer. They are widely used in improving performance of speech recognition software.

***Memory Based Models:***

An example of memory based neural network architecture is neural Turing machine. A Turing machine works by having a read write head and a tape that acts as a memory. We give a function in the form of an algorithm and then it competes that function.

A neural Turing machine is similar to Turing machine. It is built upon a Long Short Term Memory. It includes components like controller (LSTM), temporal component and memory. We have controller, addressing, read and write in it's architecture. Components of memory network are memory, input feature, updater, output feature map, responder.

**Chapter 11: Conclusion**

Sandro is asking the reader to consider some thought-provoking research questions in Deep Learning. Questions like, can we find something better than gradient descent for back propagation? Can we find better activation functions?

Overall, a great book that gives us a picture of research within A.I community, Connectionists and Good Old Fashion A.I. He says, Connectionism under the name Deep Learning is trying to take over Good Old Fashion A.I.

I really liked this example, A Sculptor has two things, a clear and precise idea what to make, skill and tools to make it. He says reach out to Philosophy for ideas. When you have no tools, reach out to mathematics. What an alluring analogy to finish the book.

An extraordinary way to finish the book. I do love philosophy and would strive to arm myself with mathematical tools in my next iterations.