First Report

Introduction

I am interested in Deep Learning. Deep learning is a subset of a subset of the way that machines think. If we want to know about Deep learning we need to have a basic idea on Artificial Intelligence and Machine Learning. So, what is AI and Machine Learning.

Artificial intelligence

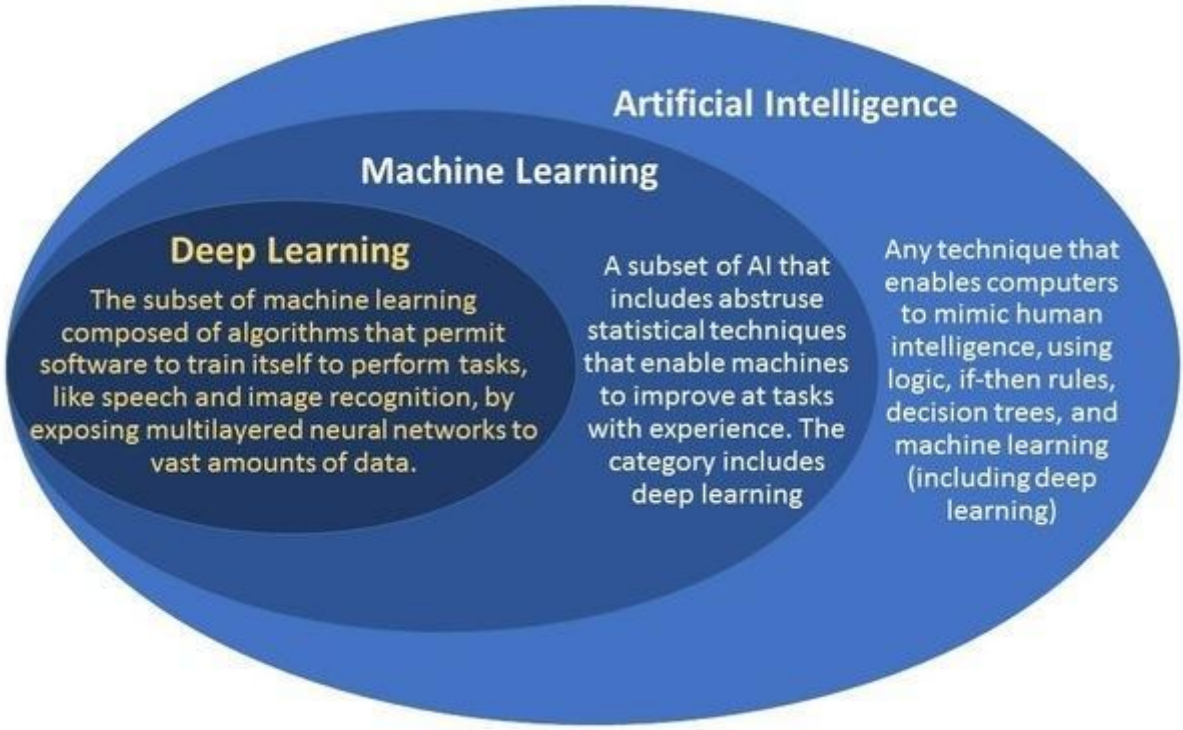
Artificial intelligence is the broadest term we use for technology that allows machines to mimic human behavior. AI can include logic, if-then rules, and more.

Machine Learning

One subset of AI technology is machine learning. Machine learning allows computers to use statistics and use the experience to improve at tasks.

Deep Learning

A subset of machine learning is deep learning, which uses algorithms to train computers to perform tasks by exposing neural nets to huge amounts of data, allowing them to predict responses without needing to actually complete every task.



Applications of Deep Learning

There are a few obvious areas where casual users are seeing a big difference in their technology experience due to deep learning. These include:

- **Speech Recognition**. Whether you use Cortana, Siri, or Ok Google/ Hey Google, your smartphone or computer's ability to recognize your voice and answer your questions is a function of deep learning. The software has been trained to recognize the ways a word can be pronounced, then understand the various types of questions that might be asked, and to provide an appropriate answer to meet your needs.

- **Smart Homes**. Using data about when your home needs light, temperature adjustments, and more, smart home technology allows customers to reduce energy usage and save money. When combined with voice recognition tools, smart homes can have profound applications for disabled consumers.
- **Image Recognition.** If you take pictures on your smartphone and store them in various cloud applications, you may have already encountered applications like Google Photos that will sort and organize albums based on various features. All the images of a particular person, for example, or images that contain a particular theme. For example, if you are using Google Lens application in your smart phone you can observe how technology is rapidly developing in which it can scan the image and say what is the name of the location based on the features it found on image

I am interested in Deep learning because, while i was reading articles and watching webinars on new technologies emerging in the world, I found a lot of new technologies are based on deep learning for example Google's search engine, voice recognition system and self-driving cars all rely heavily on deep learning. They've used deep learning networks to build a program that picks out an attractive still from a YouTube video to use as a thumbnail. In 2105 Google announced Smart Reply, a deep learning network that writes short email responses for you.

It is very interesting even to hear that machine can act like humans by training them using neural networks neural networks. So, I am very interested to learn how this process of training happens. So, I started learning neural networks. A short introduction on Neural Network

Neural Network

Neural networks are one of the most beautiful programming paradigms ever invented. In the conventional approach to programming, we tell the computer what to do, breaking big problems up into many small, precisely defined tasks that the computer can easily perform. By contrast, in a neural network we don't tell the computer how to solve our problem. Instead, it learns from observational data, figuring out its own solution to the problem at hand.

Automatically learning from data sounds promising. However, until 2006 we didn't know how to train neural networks to surpass more traditional approaches, except for a few specialized problems. What changed in 2006 was the discovery of techniques for learning in so-called deep neural networks. These techniques are now known as deep learning. They've been developed further, and today deep neural networks and deep learning achieve outstanding performance on many important problems in computer vision, speech recognition, and natural language processing. They're being deployed on a large scale by companies such as Google, Microsoft, and Facebook.

So why neural networks comes into picture even though it was there long back, because of high availability of data and computational power has been increasing exponentially just like Moore’s law. Data is everywhere. In fact, the amount of digital data that exists is growing at a rapid rate, doubling every two years, and changing the way we live. According to IBM, 2.5 billion gigabytes (GB) of data was generated every day in 2012.

An article by forbes, states that Data is growing faster than ever before and by the year 2020, about 1.7 megabytes of new information will be created every second for every human being on the planet.
I am expecting to learn as much as I can and as fast as I can, at least to the intermediate level in Neural Networks.

Time Line

After deciding that our project is on Deep Learning using keras software. I started learning the basics of Deep Learning and Neural Networks by reading the research papers and articles in the web. The tutorial CS231n, from Stanford University helped me a lot to learn the basics and I completed an online course "Fundamentals of Deep Learning" by IBM Cognative Class Online teaching Institute and another online course on Deep Learning from Udemy.
After completing the online courses, I started reading a book, Neural Networks and Deep Learning by Michael Nielsen. I completed three chapters of reading in that book and realize that It would take a lot of time to complete the book, I started working on small basic projects on deep learning using keras. Keras Documentation Kaggle , Git Hub and machine learning mastery websites really helped me a lot to learn the implementation of basic project on keras. I think one of the best way to getting started with keras is to start learning with keras documentation.

Software Tools

As we are working on Deep Learning Applications, I installed Anaconda Cloud as it has lot of inbuilt libraries like numpy, scipy, pandas, matplotlib, scikit-learn, tensor flow, keras, open cv...etc.
Anaconda Cloud Anaconda Cloud is where data scientists share their work. We can search and download popular Python and R packages and notebooks to jump start our data science work. we can also store our packages, notebooks and environments in Anaconda Cloud and share them with our team.

Python Python is a programming language that lets us work more quickly and integrate our systems more effectively. We can learn to use Python and see almost immediate gains in productivity and lower maintenance costs.
Jupyter The *Jupyter* Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.
Numpy NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code

- useful linear algebra, Fourier transform, and random number capabilities

Pandas *pandas* is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

Matplot Lib Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and Ipython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Scikit-Learn It is used for machine learning in python.

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib

Open CV OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Tensor Flow TensorFlow™ is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. Originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning and deep learning and the flexible numerical computation core is used across many other scientific domains.

Keras Keras is a high-level neural networks API, written in Python and capable of running on top of Tensoe Flow, CNTN or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Keras allows us

- For easy and fast prototyping (through user friendliness, modularity, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- To Runs seamlessly on CPU and GPU.

Projects Implemented

Predicting the onset of diabetes with in five years

As we are going to build our model based on pima Indians onset of diabetes dataset. It is better to know about the dataset.

PIMA INDIANS ONSET OF DIABETES DATASET

This is a standard machine learning dataset from the UCI Machine Learning repository. It describes patient medical record data for Pima Indians and whether they had an onset of diabetes within five years.

In this project we are going to create a model which can predict whether a person is going to get affected by diabetes within the next five years or not based on data in the Pima Indians onset of diabetes Dataset.

I implemented three layered sequential model with dense layers and used sigmoid activation function in the last layer as we are doing binary classification.

Methods used

Random seed Generator

Whenever we work with machine learning algorithms that use a stochastic process (e.g. random numbers), it is a good idea to set the random number seed. This is so that you can run the same code again and again and get the same result. This is useful if you need to demonstrate a result, compare algorithms using the same source of randomness or to debug a part of your code.

Sequential Model

The *Sequential* model is a linear stack of *layers*. This is the most widely used model while building the network.

Dense Layer

Fully connected layers are defined using the Dense class. We can specify the number of neurons in the layer as the first argument, the initialization method as the second argument as init and specify the activation function using the activation argument.

Activation Function

It is used to determine the output of neural network like yes or no. It maps the resulting values in between 0 to 1 or -1 to 1 etc. (depending upon the function).

Sigmoid Activation function

The main reason why we use sigmoid function is because it exists between (0 to 1). Therefore, it is especially used for models where we have to predict the probability as an output.Since probability of anything exists only between the range of 0 and 1, sigmoid is the right choice.

Soft max Activation Function
The soft max activation function is a more generalized logistic activation function which is used for multiclass classification.

Relu Activation Function
The ReLU is the most used activation function in the world right now.Since, it is used in almost all the convolutional neural networks or deep learning.

Compiling the model

When compiling, we must specify some additional properties required when training the network. Remember training a network means finding the best set of weights to make predictions for this problem.
We must specify the loss function to use to evaluate a set of weights, the optimizer used to search through different weights for the network and any optional metrics we would like to collect and report during training.
In this case, we will use logarithmic loss, which for a binary classification problem is defined in Keras as "binary_crossentropy". We will also use the efficient gradient descent algorithm "adam" for no other reason that it is an efficient default.
Finally, because it is a classification problem, we will collect and report the classification accuracy as the metric.

Fit the Model

We can train or fit our model on our loaded data by calling the fit() function on the model. The training process will run for a fixed number of iterations through the dataset called epochs, that we must specify using the epochs argument. We can also set the number of instances that are evaluated before a weight update in the network is performed, called the batch size and set using the batch_size argument.

Evaluate the Model

You can evaluate your model on your training dataset using the evaluate() function on your model and pass it the same input and output used to train the model.
This will generate a prediction for each input and output pair and collect scores, including the average loss and any metrics you have configured, such as accuracy.

Simple Deep Neural Network on MNIST Dataset in keras
Before going to the details of the project, it is better to know about the MNIST Dataset.

MNIST DATASET
MNIST ("Modified National Institute of Standards and Technology") is the de facto "hello world" dataset of computer vision. Since its release in 1999, this classic dataset of handwritten images has served as the basis for benchmarking classification algorithms. As new machine learning techniques emerge, MNIST remains a reliable resource for researchers and learners alike.

Each image is a 28 by 28 pixel square (784 pixels total). A standard spit of the dataset is used to evaluate and compare models, where 60,000 images are used to train a model and a separate set of 10,000 images are used to test it.
In this project I implemented five layered sequential model with dense and drop out layers to correctly identify digits from a dataset of tens of thousands of handwritten images in MNIST dataset. I used softmax activation function in the last layer as we are doing multi classification task. I achieved to get an accuracy of 98.3 % with a loss of 0.113

Handwritten Digit Recognition on MNIST Dataset using Convolutional Neural Networks in Python with Keras

This project is just to improve the accuracy of the above model by implementing the convolutional neural networks to achieve high accuracy and low loss.
In this project I implemented eight layered network with convolutional layer, Dense layer, Max-pooling layer, flatten and Dropout layer to correctly identify the digits from a dataset of tens of thousands of handwritten images in the MNIST Dataset. I used softmax activation function in the last layer as we are doing multi classification task. I got an accuracy of 99.09 % with loss of 0.0298.

Prediction of Cancer based on the person's details

In this model, I need to predict whether the person or individual is going to get cancer or not based on the individual medical details in the dataset provided in kaggle.
As I found some useless features in the dataset, I cleaned the dataset by removing the useless features. After that, I implemented three layered sequential neural network with dense layers to predict whether the person is going to get effected by cancer or not. I used sigmoid activation function in the last layer as we are doing binary classification. After prediction , I got an accuracy of 91.85%.

Predict Sentiment From Movie Reviews Using Deep Learning on IMDB Dataset

Before going to details of this project it is better to know about sentiment analysis and IMDB Dataset.
SENTIMENT ANALYSIS
In short, sentiment analysis is a natural language processing problem where text is understood and the underlying intent is predicted.
IMDB DATASET
The Large Movie Review Dataset (often referred to as the IMDB dataset) contains 25,000 highly polar moving reviews (good or bad) for training and the same amount again for testing. The problem is to determine whether a given movie review has a positive or negative sentiment.
In this project we need to predict the sentiment of movie reviews as either positive or negative in Python using the Keras deep learning library. I used word embedding technique to label the words with numbers as neural network expects only numerical data as an input. I performed six layered sequential model with embedding layer, one dimensional convolutional layer, max pooling layer, Flatten and Dense layer. I used

sigmoid activation function in the last layer as we are doing binary classification. I got an accuracy of 87.79 %.

New methods used

Word Embeddings

A recent breakthrough in the field of natural language processing is called word Embeddings.

This is a technique where words are encoded as real-valued vectors in a high-dimensional space, where the similarity between words in terms of meaning translates to closeness in the vector space.

Discrete words are mapped to vectors of continuous numbers. This is useful when working with natural language problems with neural networks and deep learning models are we require numbers as input.

Keras provides a convenient way to convert positive integer representations of words into a word embedding by an Embedding Layer

The layer takes arguments that define the mapping including the maximum number of expected words also called the vocabulary size (e.g. the largest integer value that will be seen as an integer). The layer also allows you to specify the dimensionality for each word vector, called the output dimension.

We would like to use a word embedding representation for the IMDB dataset.

Let's say that we are only interested in the first 5,000 most used words in the dataset. Therefore our vocabulary size will be 5,000. We can choose to use a 32-dimension vector to represent each word. Finally, we may choose to cap the maximum review length at 500 words, truncating reviews longer than that and padding reviews shorter than that with 0 values.

Cat and dog classifier

In this project I am going to create a model which can say whether the given input is cat or dog. I implemented nine layered sequential model with convolution, activation, max polling, drop out and dense layers to build the model. I used the dataset available in the kaggle which contains both cats and dogs with labels to train the model. I used sigmoid activation function in the last layer as we are doing binary classification. I achieved an accuracy of 85 %.

Next steps proposed

I found two interesting projects in github . They are

> **Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow.**
>
> This is an implementation of Mask R CNN on Python, Keras, and TensorFlow. The model generates bounding boxes and segmentation masks for each instance of an object in the image. It's based on Feature Pyramid Network (FPN) and a ResNet101 backbone.

Real-time face detection and emotion/gender classification using fer2013/imdb datasets with a keras CNN model and openCV.

In this project, we are going to implement a general convolutional neural network framework for designing real time CNNs. We will validate our model by creating a real time vision system which accomplishes the task of face detection, gender classification and emotion classification simultaneously in one blended step using CNN architecture.

Please guide me with the interesting projects.

Biblography

The resources I used for this report are

- Keras documentation
- Machinelearningmastery.com
- Github
- Kaggle
- Articles from simplilearn.com
- Wikipedia
- Neural networks and deep learning book by michael nielsen