

## INTRODUCTION TO PYTHON

Python is an interpreted high-level programming language for general purpose programming. It is initially designed by 'Guido Van Rossum' in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis code readability, and its syntax allows programmers to express concepts in fewer lines of code.

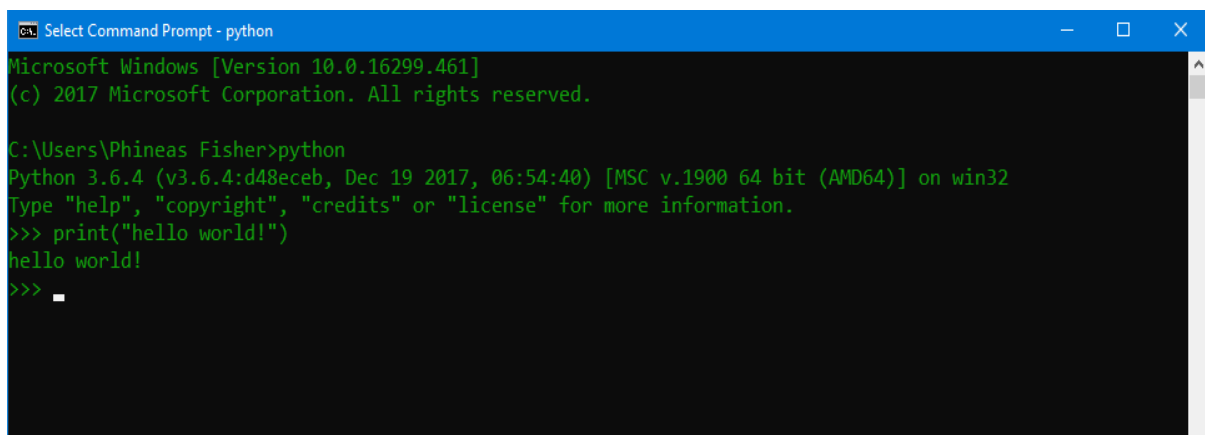
Python source code is also available under the GNU General Public License (GPL). Python is named after a TV Show called '**Monty Python**' and not after Python-the snake.

There are two major Python versions – **Python 2 and Python 3**. Both are quite different. Python 3.0 was released in 2008. Although this version is supposed to be backward incompatibles, later on many of its important features have been backported to be compatible with version 2.7.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python is **interpreted**. Python is processed at runtime by the interpreter. You do not need to compile your program before executing it.

Python is **Interactive**. You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.



```
Select Command Prompt - python
Microsoft Windows [Version 10.0.16299.461]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Phineas Fisher>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hello world!")
hello world!
>>> _
```

### **PYTHON FEATURES**

Python includes many features. Some are listed below: -

- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **A broad standard library** - Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** - Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Databases** - Python provides interfaces to all major commercial databases.
- **GUI Programming** - Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of UNIX.
- **Scalable** - Python provides a better structure and support for large programs than shell scripting.

Apart from above-mentioned features, Python has a big list of good features. A few are listed below –

- It can be used as a **scripting language** or can be **compiled to byte-code** for building large applications.
- It can be **easily integrated** with C, C++, COM, ActiveX, CORBA, and Java.
- It supports **automatic garbage collection**.

## INTRODUCTION TO KERAS

Two of the top numerical platforms in Python that provide the basis for Deep Learning research and development are **Theano** and **TensorFlow**. Both are very powerful libraries, but both can be difficult to use directly for creating deep learning models.

Keras is the Python library that provides a clean and convenient way to create a range of deep learning models on top of Theano or TensorFlow.

### **What is Keras?**

Keras is a minimalist Python library for deep learning that can run on top of Theano or TensorFlow.

It was developed to make implementing deep learning models as fast and easy as possible for research and development.

It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks. It is released under the permissive MIT license.

Keras was developed and maintained by **François Chollet**, a Google engineer using four guiding principles:

- **Modularity:** A model can be understood as a sequence or a graph alone. All the concerns of a deep learning model are discrete components that can be combined in arbitrary ways.
- **Minimalism:** The library provides just enough to achieve an outcome, no frills and maximizing readability.
- **Extensibility:** New components are intentionally easy to add and use within the framework, intended for researchers to trial and explore new ideas.
- **Python:** No separate model files with custom file formats. Everything is native Python.

The focus of Keras is the idea of a **model**. The main type of model is called a **Sequence** which is a linear stack of layers. You create a sequence and add layers to it in the order that you wish for the computation to be performed.

Models can be instantiated using the `Sequential()` class. Neural networks are built up from bottom layer to top using `add()` method.

### **ADVANTAGES OF KERAS**

Keras has many advantages over other deep learning libraries. Some advantages are as follows:

- **Rapid Prototyping** - If you want to quickly build and test a neural network with minimal lines of code, choose Keras. With Keras, you can build simple or very complex neural networks within a few minutes. The Model and the Sequential APIs are so powerful that you can do almost everything you may want.
- **Functionality** – Keras provides all the general purpose functionalities for building Deep learning models.
- **Library** - Keras library is very flexible, constantly being updated and being further integrated with tensorflow.
- **Integration** – It seamlessly integrates with tensorboard: A visualisation tool for neural network learning and debugging.
- **Speed** – It's having a high running speed.

## **INTRODUCTION TO TENSORFLOW**

**TensorFlow** is an open-source software library. **TensorFlow** was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

TensorFlow is an interface for expressing machine learning algorithms, and an implementation for executing such algorithms.

**TensorFlow** is basically a software library for numerical computation using **data flow graphs** where:

- **Nodes** in the graph represent mathematical operations.
- **Edges** in the graph represent the multidimensional data arrays (called **tensors**) communicated between them.

This flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API (Application Programming Interface).

### **TensorFlow APIs**

**TensorFlow** provides multiple APIs. These can be classified into 2 major categories:

- Low level API:
  - Complete programming control
  - Recommended for machine learning researchers
  - Provides fine levels of control over the models
  - **TensorFlow Core** is the low level API of TensorFlow.
- High level API:
  - Built on top of **TensorFlow Core**
  - Easier to learn and use than **TensorFlow Core**
  - Make repetitive task easier and more consistent between different users.
  - **Tf.contrib.learn** is an example of a high level API.

## NUMPY

NumPy is the fundamental package for scientific computing with Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. 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

NumPy provides support for large multidimensional arrays and matrices along with a collection of mathematical functions to operate on these elements. The project relies on well-known packages implemented in another languages (like FORTRAN) to perform efficient computations, bringing the user both the expressiveness of Python and a performance similar to MATLAB or FORTRAN.

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

### Array

A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the *rank* of the array; the *shape* of an array is a tuple of integers giving the size of the array along each dimension.

NumPy also provides many functions to create arrays:

```
import numpy as np

a = np.zeros((2,2)) # Create an array of all zeros
print(a)

c = np.full((2,2), 7) # Create a constant array
print(c)             # Prints [ [ 7.  7.] [ 7.  7.]]

e = np.random.random((2,2)) # Create an array filled with random values
```

## DIGIT RECOGNIZER

---

```
print(e)
```

### **TENSORFLOW VS NUMPY**

Few people make this comparison, but TensorFlow and NumPy are quite similar. Both are N-d array libraries. But still they are differ.

<b>TensorFlow</b>	<b>NumPy</b>
TensorFlow tends to be more ML (machine learning) or Deep Learning library that helps building heuristics, and statistical data analysis through a longer periods of time.	NumPy is more general purpose high level mathematical library used to work with models on the fly.
It does it by taking various inputs, learning from various sources, and compiling the best possible scenario of execution.	It can be used to work with some insane data sets, and get some real-time results.
TensorFlow would learn from that data, and would use some advanced algorithms such as multiple linear regression, interval estimation or significance, in order to generate the best possible scenario for the given inputs.	NumPy helps you create data and send it to TensorFlow.
The main focus of the library is to provide an easy-to-use API to implement practical machine learning algorithms and deploy them to run on CPUs, GPUs, or a cluster.	Numpy is the fundamental package for numerical computation in python.
Coders can use either C++ or <b>Python</b>	This package can only be used from Python.

These are the some major differences in both libraries.



### **INTRODUCTION TO FLASK**

Flask is a small and powerful web framework for Python. Flask is written in Python. Flask can be used for building simple as well as complex, database-driven websites.

To understand what is Flask, let's first consider the definition of a framework.

#### **Def: Framework**

A framework in coding is a set of classes, functions, and variables that form a mindset for thinking about how to solve a given set of challenges within a specific problem domain.

With this definition in hand, you can define a web development framework as a set of classes, functions, and variables that form a mindset for thinking about how to solve challenges around building websites.

The name is intended to conjure the imagery of a small Flask of water - because it requires only a little "water" to get things done. Flask is also referred to as a microframework, since you need very little knowledge to use Flask correctly. However, there are plenty of extensions and advanced features that allow it to be used as much more.

### **ARTIFICIAL INTELLIGENCE**

**Artificial Intelligence** (AI also Known as **Machine Intelligence** MI) is intelligence demonstrated by machines, in contrast to the **natural intelligence** (NI) displayed by humans and other animals.

#### **What is Artificial Intelligence?**

According to the father of Artificial Intelligence, John McCarthy, it is “The science and engineering of making intelligent machines, especially intelligent computer programs”.

Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

#### **Goals of AI**

- **To create Expert Systems** – The systems which exhibit intelligent behaviour, learn, demonstrate, explain, and advice its users.
- **To Implement Human Intelligence in Machines** - Creating systems that understand, think, learn, and behave like humans.

### **TYPES OF ARTIFICIAL INTELLIGENCE**

With advances in computing power – including machine learning, neural networks, natural language processing, genetic algorithms and computational creativity, to name just a few – it increasingly seems likely that artificial intelligence is evolving from simple to self-aware machines. Here is a look at where AI is now:

#### ➤ **Machine Learning**

Machine Learning is where a computer system is fed large amounts of data, which it then uses to learn how to carry out a specific task, such as understanding speech or explaining a photograph.

#### ➤ **Artificial Neural Network**

Neural Networks are the key to process "Machine Learning". Researchers tried to reproduce the workings of the human brain by creating artificial networks of neurons and synapses.

#### ➤ **Big Data**

Big Data Large scale data analysis, often called "big data," harnesses the power of many computers to discover facts and relations in data that the human mind cannot comprehend. Trillions of credit card charges or billions of social network relations can be scanned and correlated using a variety of statistical methods to discover useful information.

### **MACHINE LEARNING**

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. **The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly.

### **MACHINE LEARNING METHODS**

Machine learning algorithms are often categorized as supervised or unsupervised.

- **Supervised machine learning algorithms** can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.
- In contrast, **unsupervised machine learning algorithms** are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data.
- **Semi-supervised machine learning algorithms** fall somewhere in between supervised and unsupervised learning, since they use both labelled and unlabelled data for training – **typically a small amount of labelled data and a large amount of unlabelled data**. The systems that **use this method are able to considerably improve learning accuracy**. Usually, semi-supervised learning is chosen when the acquired labelled data requires skilled and relevant resources in order to train it / learn from it.

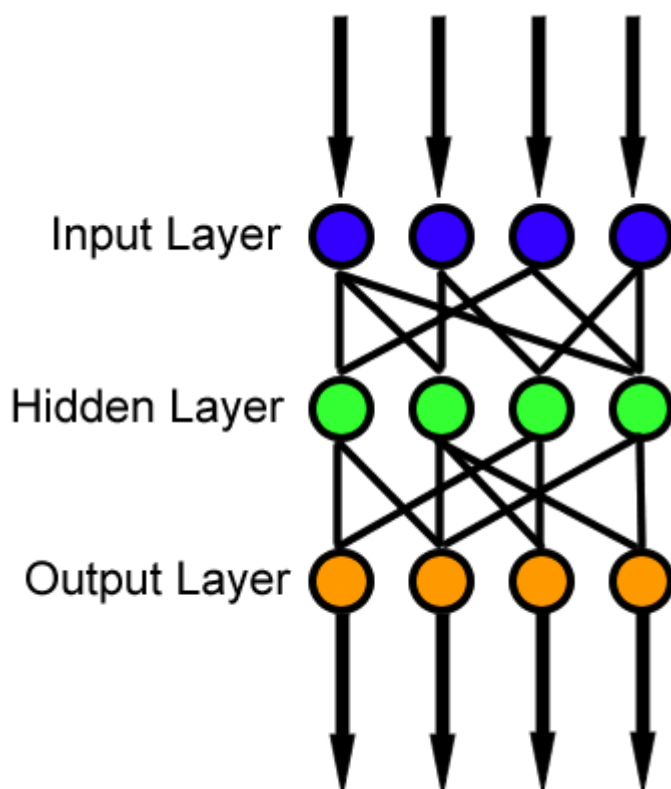
Machine learning enables analysis of massive quantities of data. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it may also require additional time and resources to train it properly. Combining machine learning with AI and cognitive technologies can make it even more effective in processing large volumes of information.

## NEURAL NETWORK

An Artificial Neural Network (ANN), popularly known as ‘**Neural Network**’ is a computational model that is inspired by the way biological neural networks in the human brain process information. Artificial Neural Networks have generated a lot of excitement in Machine Learning research and industry, thanks to many breakthrough results in **speech recognition, computer vision** and **text processing**.

Basically, there are 3 different layers in a neural network: -

1. **Input Layer** (All the inputs are fed in the model through this layer)
2. **Hidden Layers** (There can be more than one hidden layers which are used for processing the inputs received from the input layers)
3. **Output Layer** (The data after processing is made available at the output layer)



## **Input Layer**

The Input layer communicates with the external environment that presents a pattern to the neural network. Its job is to deal with all the inputs only. This input gets transferred to the hidden layers which are explained below.

Every input neuron should represent some independent variable that has an influence over the output of the neural network.

## **Hidden Layer**

The hidden layer is the collection of neurons which has activation function applied on it and it is an intermediate layer found between the input layer and the output layer. Its job is to process the inputs obtained by its previous layer. So it is the layer which is responsible extracting the required features from the input data.

Many researches has been made in evaluating the number of neurons in the hidden layer but still none of them was successful in finding the accurate result. Also there can be multiple hidden layers in a Neural Network.

If the number of neurons are **less** as compared to the complexity of the problem data then there will be very few neurons in the hidden layers to **adequately detect the signals** in a complicated data set. If unnecessary **more neurons** are present in the network then **Overfitting** may occur.

Several methods are used till now which do not provide the exact formula for calculating the number of hidden layer as well as number of neurons in each hidden layer.

## **Output Layer**

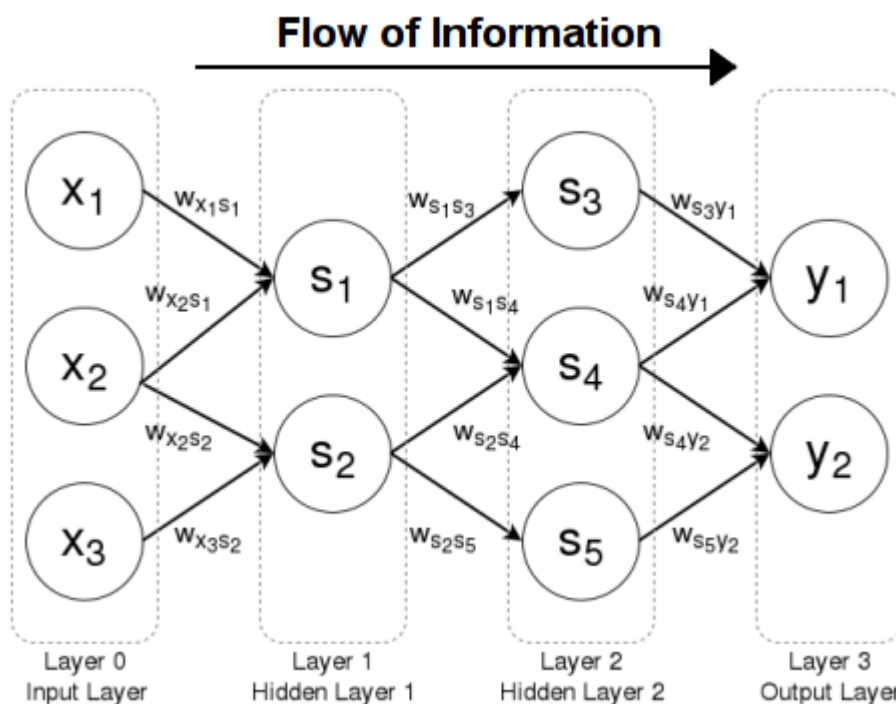
The output layer of the neural network collects and transmits the information accordingly in way it has been designed to give. The pattern presented by the output layer can be directly traced back to the input layer. The number of neurons in output layer should be directly related to the type of work that the neural network was performing. To determine the number of neurons in the output layer, first consider the intended use of the neural network.

## TYPES OF NEURAL NETWORK

There are several kinds of artificial neural networks. These type of networks are implemented based on the mathematical operations and a set of parameters required to determine the output. Let's look at some of the neural networks:

### 1) FeedForward Neural Network – Artificial Neuron:

Application of Feed forward neural networks are found in computer vision and speech recognition where classifying the target classes are complicated. These kind of Neural Networks are responsive to noisy data and easy to maintain.



### 2) Recurrent Neural Network – Long Short Term Memory:

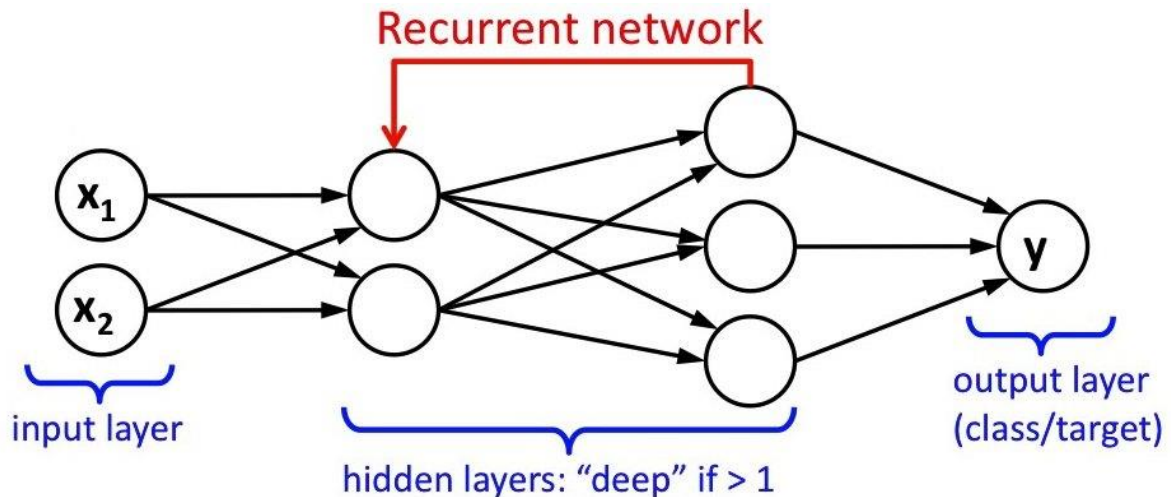
The Recurrent Neural Network works on the principle of saving the output of a layer and feeding this back to the input to help in predicting the outcome of the layer.

Here, the first layer is formed similar to the feed forward neural network with the product of the sum of the weights and the features. The recurrent neural network process starts once this is computed, this means that from one time step to the next each neuron will remember some information it had in the previous time-step. This makes each neuron act like a memory cell in performing computations.



In this process, we need to let the neural network to work on the front propagation and remember what information it needs for later use. Here, if the prediction is wrong we use the learning rate or error correction to make small changes so that it will gradually work towards making the right prediction during the back propagation.

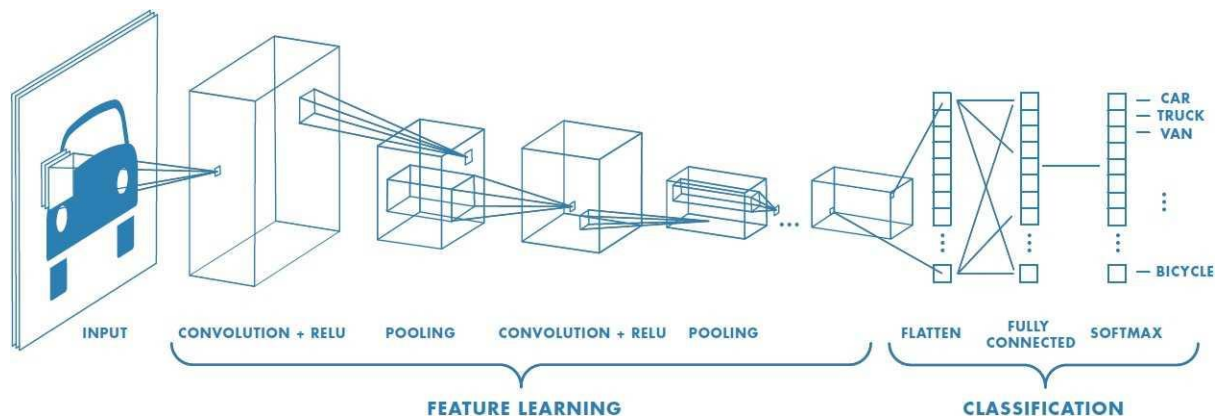
This is how a basic Recurrent Neural Network looks like,



### 3) Convolutional Neural Network:

Convolutional neural networks are similar to feed forward neural networks, where the neurons have learn-able weights and biases. Its application have been in signal and **image processing** which takes over OpenCV in field of **computer vision**.

Below is a representation of a ConvNet, in this neural network, the input features are taken in batch wise like a filter. This will help the network to remember the images in parts and can compute the operations. These computations involve conversion of the image from RGB or HSI scale to Grey-scale. Once we have this, the changes in the pixel value will help detecting the edges and images can be classified into different categories.

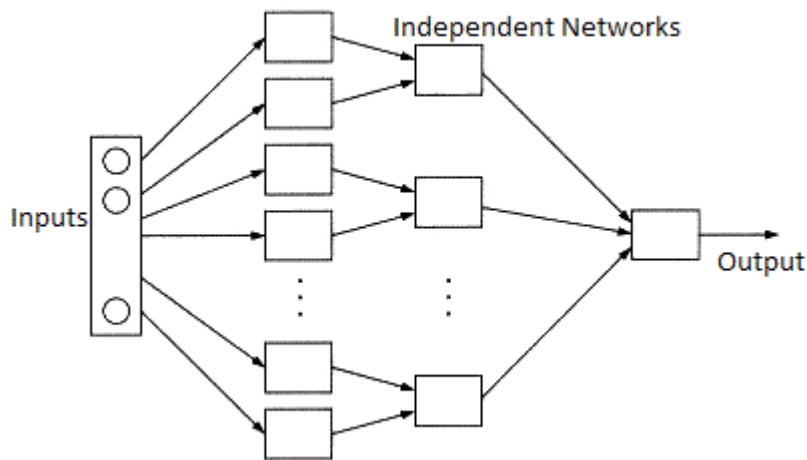


ConvNet are applied in techniques like signal processing and image classification techniques. Computer vision techniques are dominated by convolutional neural networks because of their accuracy in image classification.

#### 4) **Modular Neural Network:**

Modular Neural Networks have a collection of different networks working independently and contributing towards the output. Each neural network has a set of inputs which are unique compared to other networks constructing and performing sub-tasks. These networks do not interact or signal each other in accomplishing the tasks. The advantage of a modular neural network is that it breakdowns a large computational process into smaller components decreasing the complexity. This breakdown will help in decreasing the number of connections and negates the interaction of these network with each other, which in turn will increase the computation speed. However, the processing time will depend on the number of neurons and their involvement in computing the results.

Below is a visual representation,



Modular Neural Networks (MNNs) is a rapidly growing field in artificial Neural Networks research. The **different motivations for creating** MNNs are: biological, psychological, hardware, and computational.

### **CONVOLUTIONAL NEURAL NETWORK**

Convolutional neural networks (or convnets for short) are used in situations where data can be expressed as a "map" wherein the proximity between two data points indicates how related they are. An image is such a map, which is why you so often hear of convnets in the context of image analysis. If you take an image and randomly rearrange all of its pixels, it is no longer recognizable. The relative position of the pixels to one another, that is, **the order, is significant.**

A ConvNet takes an image expressed as an array of numbers, applies a series of operations to that array and, at the end, returns the probability that an object in the image belongs to a particular class of objects. For instance, a ConvNet can let you know the probability that a photo you took contains a building or a horse or what have you. It might be used to distinguish between very similar instances of something.

Convnets contain **one or more** of each of the **following layers**:

- Convolutional layer
- ReLU (Rectified linear units) layer
- Pooling layer
- Fully connected layer
- Regularization

#### **Convolutional Layer**

The architecture of a convnet is modelled after the mammalian visual cortex, the part of the brain where visual input is processed. Within the visual cortex, specific neurons fire only when particular phenomena are in the field of vision. One neuron might fire only when you are looking at a left-sloping diagonal line and another only when a horizontal line is in view.

Our brains process images in layers of increasing complexity. The first layer distinguishes basic attributes like lines and curves. At higher levels, the brain recognizes that a configuration of edges and colours is, for instance, a house or a bird.

In a similar fashion, a convnet processes an image using a matrix of weights called filters (or features) that detect specific attributes such as diagonal edges, vertical edges, etc. Moreover, as the image progresses through each layer, the filters are able to recognize more complex attributes.

To a computer, an image is just an array of numbers. An image using the full spectrum of colours is represented using a 3-dimensional matrix of numbers.

The depth of the matrix corresponds to RGB values. For the sake of simplicity, we'll initially consider only grayscale images. Each pixel in a grayscale image can be represented using a single value that indicates the intensity of the pixel. These values lie between 0 and 255, where 0 is black and 255 is white.

The convolution layer is always the first step in a convnet. Let's say we have a 10 x 10 pixel image, here represented by a 10 x 10 x 1 matrix of numbers:

0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	1	1
0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0

The **output** of a convolution layer is something called a **feature map** (or activation map).

In order to generate a feature map, we take an array of weights (which is just an array of numbers) and slide it over the image, taking the dot product of the smaller array and the pixel values of the image as we go. This operation is called convolution.

The array of weights is referred to as a filter or a feature. Below, we have a 3 x 3 filter (as with the image, we've used 1s and 0s for simplicity).

1	0	0
0	1	0
0	0	1

We then use the filter to generate a feature map

Image

0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	1	1
0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0

Filter

1	0	0
0	1	0
0	0	1

Feature Map

0	0	0	0	1	0	1	0
2	0	0	0	0	1	0	1
0	3	0	0	0	1	2	1
0	0	3	0	0	0	1	1
0	0	0	2	0	0	0	1
0	0	0	0	2	0	0	0
2	0	0	0	0	2	0	0
0	3	0	0	0	0	2	0

$$0 \times 1 + 0 \times 0 + 0 \times 0 + 0 \times 0 + 0 \times 1 + 0 \times 0 + 0 \times 0 + 1 \times 0 + 0 \times 1 = 0$$

Filters are used to find where in an image details such as horizontal lines, curves, colours, etc. occur. The filter above finds right-sloping diagonal lines.

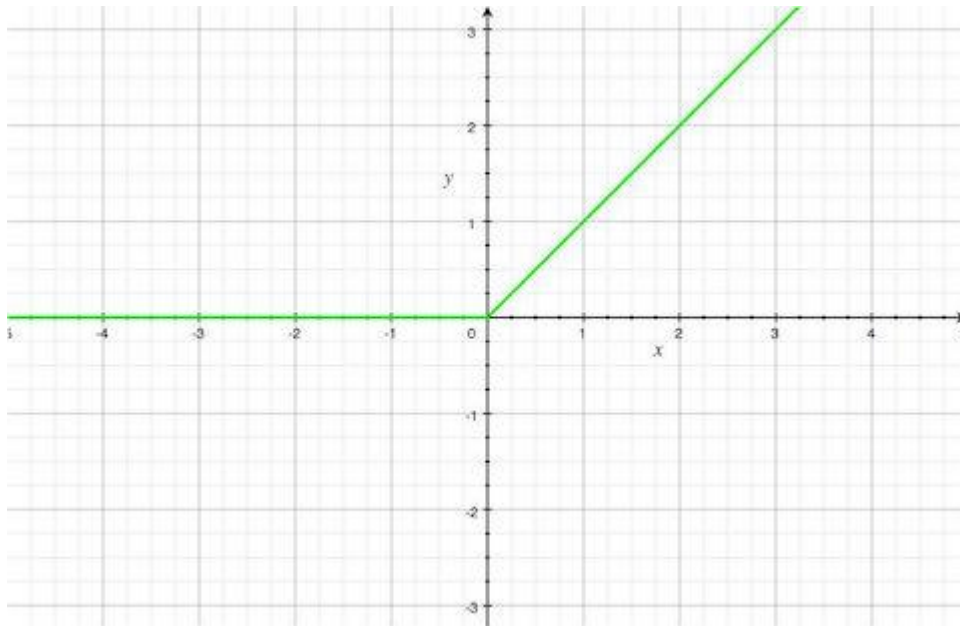
## ReLU Layer

The ReLU (short for rectified linear units) layer commonly follows the convolution layer. ReLU keep the math from breaking by turning all negative numbers to 0) a stack of images becomes a stack of images with no negative values. The addition of the ReLU layer allows the neural network to account for non-linear relationships, i.e. the ReLU layer allows the convnet to account for situations in which the relationship between the pixel value inputs and the convnet output is not linear. Note that the convolution operation is a linear one.

The ReLU function takes a value  $x$  and returns 0 if  $x$  is negative and  $x$  if  $x$  is positive.

X is the value of pixels.

$$f(x) = \max(0, x)$$



As you can see from the graph, the ReLU function is nonlinear. In this layer, the ReLU function is applied to each point in the feature map. The result is a feature map without negative values.

## ReLU Layer

**Filter 1 Feature Map**

9	3	5	-8
-6	2	-3	1
1	3	4	1
3	-4	5	1



9	3	5	0
0	2	0	1
1	3	4	1
3	0	5	1

Other functions such as tanh or the sigmoid function can be used to add non-linearity to the network, but ReLU generally works better in practice.

### **Pooling Layer**

The pooling layer also contributes towards the ability of the convnet to locate features regardless of where they are in the image. In particular, the pooling layer makes the convnet less sensitive to small changes in the location of a feature, i.e. it gives the convnet the property of translational invariance in that the output of the pooling layer remains the same even when a feature is moved a little. Pooling also reduces the size of the feature map, thus simplifying computation in later layers.

There are a number of ways to implement pooling, but the most effective in practice is max pooling. To perform max pooling, imagine a window sliding across the feature map. As the window moves across the map, we grab the largest value in the window and discard the rest.

The most common implementation of max pooling, and the one used I used in my project model, uses a 2 x 2 pixel window, i.e. we take the largest value in the window, move the window over by 2 pixels, and repeat.

### **Fully-Connected Layer**

The fully-connected layer is where the final "decision" is made. At this layer, the convnet returns the probability that an object in a photo is of a certain type.

The convolutional neural network which has been used in this project implements something called '**Supervised Learning**'. In supervised learning, a neural network is provided with labelled training data from which to learn.

Let's say you want your convnet to tell you if an image is of a cat or of a dog. You would provide your network with a large set of pictures of cats and dogs, where pictures of cats are labelled 'cat' and pictures of dogs are labelled 'dog'. This is called the **training set or dataset**.

Then, based on the difference between its guesses and the actual values, the network adjusts itself such that it becomes more accurate each time you run a test image through it.

You confirm that your network is in fact able to properly classify photos of cats and dogs in general (as opposed to just being able to classify photos in the training set you provided) by running it against an unlabelled collection of images. This collection is called the **test set**.

The fully-connected layer has at least 3 parts - an input layer, a hidden layer, and an output layer. The input layer is the output of the preceding layer, which is just an array of values.

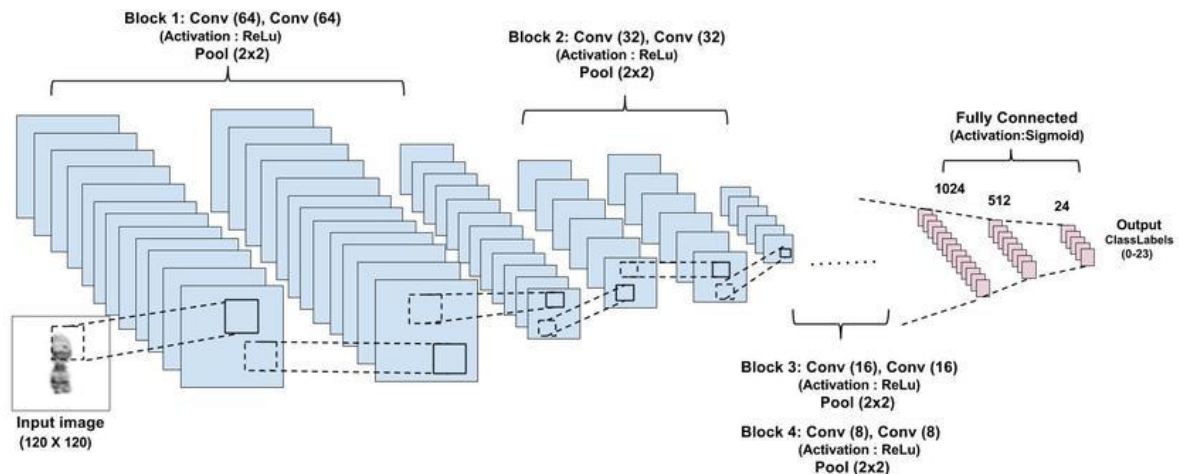
## Hyperparameters

It should be said at this point that each of these layers (with the exception of the loss layer) can be multiply stacked on one another. You may very well have a network that looks like this:

**Convolutional layer >> ReLU >> pooling >> convolutional layer >> pooling >> fully-connected layer >> convolutional layer >> fully-connected layer >> loss layer**

There are other parameters in addition to the order and number of the layers of a convnet that an engineer can modify. The parameters that are adjusted by a human agent are referred to as hyperparameters. This is where an engineer gets to be creative.

Other hyperparameters include the size and number of filters in the convolution layer and the size of the window used in the max pooling layer.





### **FEASIBILITY STUDY**

Many feasibility studies are disillusioning for both users and analysts. First, the study often pre-supposes that when the feasibility document is being prepared, the analyst is in a position to evaluate solutions. Second, most studies tend to overlook the confusion inherent in system development-the constraints and the assumed attitudes. If the feasibility study is to serve as a decision document, it must answer three key questions:

- Is there a new and better way to do the job that will benefit the user?
- What are the cost and savings of the alternative(s)?
- What is recommended?

The most successful system projects are not necessarily the biggest or the most visible in a business but rather those that truly meet user expectations. More projects fail because of inflated expectation than for any other reason.

## **Feasibility Considerations**

The three considerations involved in the feasibility analysis: economic, technical and behavioural.

### **Economic feasibility**

Economic analysis is the most frequently used method for evaluating the effectiveness of the candidate system, commonly known as cost/benefit analysis. This procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. Otherwise, further justification or alterations in the proposed system will have to be made if it is to have a chance of being approved. This is an ongoing effort that improves in an accuracy at each phase of the system life cycle.

### **Technical feasibility**

Technical feasibility centres on the existing computer system (hardware, software, etc.) And to what extent it can support the proposed addition. For instance, if the current computer is operating at 80% capacity-an arbitrary ceiling-then running another application could overload the system or require additional hardware. This involves financial consideration to accommodate technical enhancements. If the budget is a serious constraint, then the project is judged not feasible. Management system is technically feasible because it is memory efficient, it require less memory space. It will neither lead to overloading nor does it require any additional hardware.

### **Behavioural feasibility**

People are inherently resistant to change, and computers have been known to facilitate change. An estimate has been made of how strong a reaction the user is likely to have towards the development of a computerized system. It is common knowledge that computer installation has something to do with turnover, transfers, retraining and changes in employee job status. Therefore, it is understandable that the introduction of a candidate system requires a special effort to educate, develop, and train the staff.

Digit Recognizer is behavioural feasible because of its user-friendly nature. It is made in an interactive way to ease the user in using this system. A novice user with minimal guidance can use it. Thus, training time and cost of the user is reduced. This system is flexible to; further

changes can be done easily. Data related changes can be made easily. Thus increases the life of this system and reduces the related cost.

### **REQUIREMENT ANALYSIS**

REQUIREMENT ANALYSIS is the first technical step in the software process. It is at this point that a general statement of software scope is refined into a concrete specification that becomes the foundation of all software engineering activities that follow. Software requirement engineering is a process of discovery, refinement, modelling and specification. The system requirements and the role allocated to the software, initially established by the system engineer are refined in detail. Models in the required data, information and control flow and operational behaviour are created. Both the developer and customer have to take an active part in software requirement engineering. Requirement analysis is a software engineering task that bridges the gap between the customer and the developers. Facilitation Application Specification Techniques (FAST) was applied at the time of requirement analysis as the developer and the customer or the user worked together as a unit. The success of a system depends largely on how accurately a problem is defined, thoroughly investigated, and properly carried out through the choice of solution. User requirement analysis and need identification are concerned with what the user needs rather than what he/she wants. Not until the problem has been identified, defined, & evaluated should the analyst think about solutions and whether the problem is worth solving. This step is intended to help the user and the analyst understand the real problem rather than its symptoms.

What kind of information do we require?

Before one determines where to go for information or what tools to use, the first requirement is to figure out what information to gather. Much of the information we need to analyse relates to the organization in general, like knowledge about the people who run the present system-their job functions and information requirements the relationship of their job to their existing system. System should be interactive & users friendly so that the training period for users should be less & they can easily learn & use our system, because our job is to reduce their complexity & workload not to increase it. The data related to the employees changes frequently because of retirements, deputations, trainings and new recruitments. So, the system should be flexible so that further changes without affecting the current data can be done easily.

Strategies for determining information requirements are:

There are two key strategies or general approaches for eliciting information regarding the user's requirements: (1) Asking, (2) Getting information from the existing information system.

This strategy obtains information from users by simply asking them about the requirements. It assumes a stable system where users are well informed and can overcome biases in defining their problem.

- What kind of information is required?
- What fields should be included in the module?
- What are the qualification for particular designation?
- What all information they want from the program?
- What information they want to be automatically generated?
- Different categories of the departments?

Brain storming is a technique used for generating new ideas and obtaining general information requirement. This method is appropriate for eliciting non-conventional solutions to problems. A guided approach to brain storming asks people involved in the project to define ideal solutions and then select best feasible one. It works well for users who have system knowledge but have difficulty accepting new ideas.

Then with the help of the different methods of feasibility the best or the most feasible approach is taken. This debate is continued until participants responses have converged enough. This method has an advantage over brainstorming in that participants are not subjected to psychological pressure from others with presumed authority or influence.

### **Getting information from the existing information system**

Determining information from an existing application has been called the data analysis approach. It simply asks the user what information is currently received and what other information is required. It relies heavily on the user to articulate information needs. The analysts examine all reports, discusses with the user each piece of information examined, and determines unfulfilled information needs by interviewing the user. The analyst is primarily involved in improving the existing flow of data to the user. In contrast to this method is decision analysis. This breaks down a problem into parts, which allows the user to focus separately on the critical issues. It also determines policy and organizational objectives relevant to the decision areas identified and the specific steps required to complete each major decision. Then the analyst and the user refined the decision process and the information requirements for a final statement of information requirements. The data analysis method is ideal for making structured decisions, although it requires that users articulate their information requirements.

A major drawback is a lack of established rules for obtaining and validating information needs that are not linked to organizational objectives.

In the decision analysis method, information needs are clearly linked to decision and organizational objectives. It is useful for unstructured decisions and information tailored to the user's decision-making style. The major drawback, though, is that information requirements may change when the user is promoted or replaced.

### **SOFTWARE PLANNING AND DESIGN**

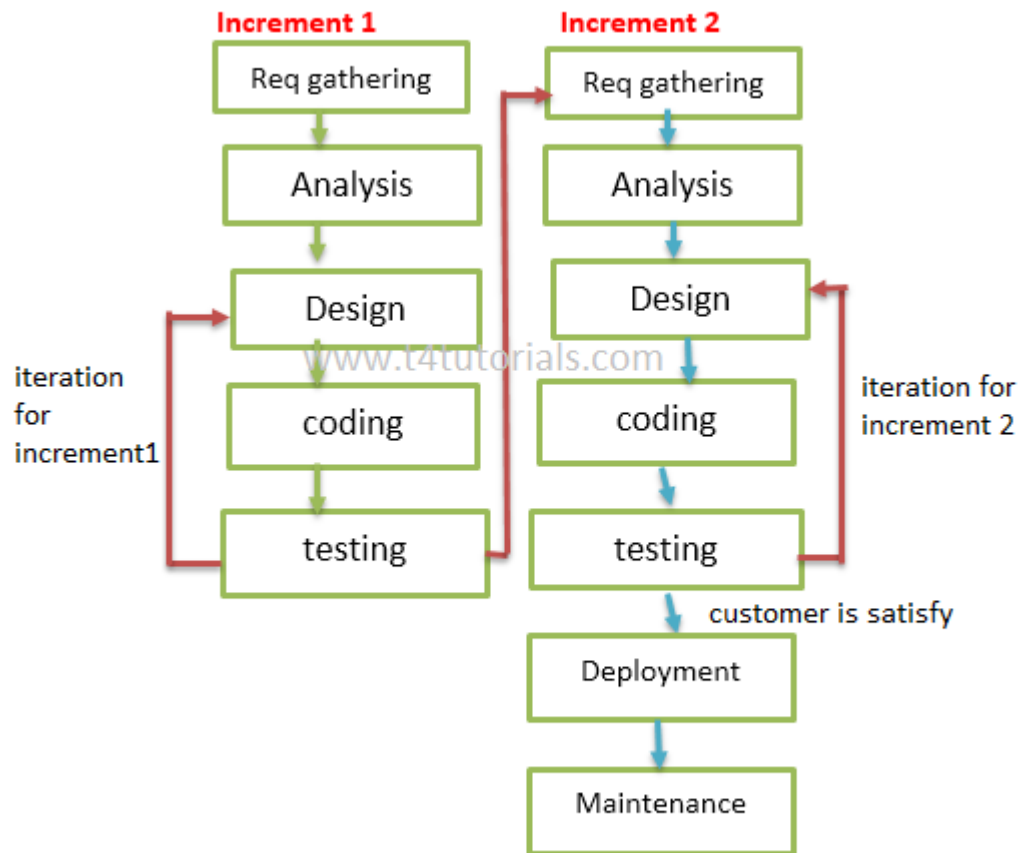
Software planning and design is an important stage in the development of software. First of all a model needs to be selected for developing software. A process model for software engineering is chosen based on the nature of the project and application, the method and tools to be used, and the controls for deliverables that are required. Keeping in mind the nature of the project, its application and the language (Python) on which the DEEP NEURAL NETWORK was supposed to be developed on Evolutionary model was selected.

Evolutionary models have been explicitly designed to accommodate a product that involves modifications over a time. Evolutionary models are iterative. They are characterized in a manner that enables software engineers to develop increasingly more complete version of the software. Among the versions evolutionary models INCREMENTAL MODEL was used.

The incremental model of the linear sequential model with the iterative philosophy of prototyping. The incremental model applies linear sequence in a staggered fashion. Referring to the figure there are four steps in each increment.

- **Software requirement analysis** -- The requirement gathering process is intensified and focused specially on software. To understand the nature of the program to be built, the software engineer must understand the information domain for the software as well as the required function behaviour, performance and interface. Requirements for both the system and the software are documented and revived with the customer.
- **Design:** -- Software design is actually a multistep process that focuses on four distinct attributes of the program, data structure, software architecture, and interface representation and procedure retail. The design process translates requirements into a presentation of the software that can be accessed for quality before coding begins. Like requirements, the design is documented and becomes part of the software configuration.
- **Code generation:** -- The design must be translated into a machine-readable form. The code generation step performs this task.
- **Testing:** -- Once software has been generated testing begins. The testing processes focuses on the logical internals of the software, ensuring that all the statements have been tested, and on the functional external; that is, conducting test to uncover errors and ensure that defined input will produce actual results that agree with expected results. The first increment of the incremental model produced the core product i.e. basic requirements were addressed but many supplementary features remain

undelivered.



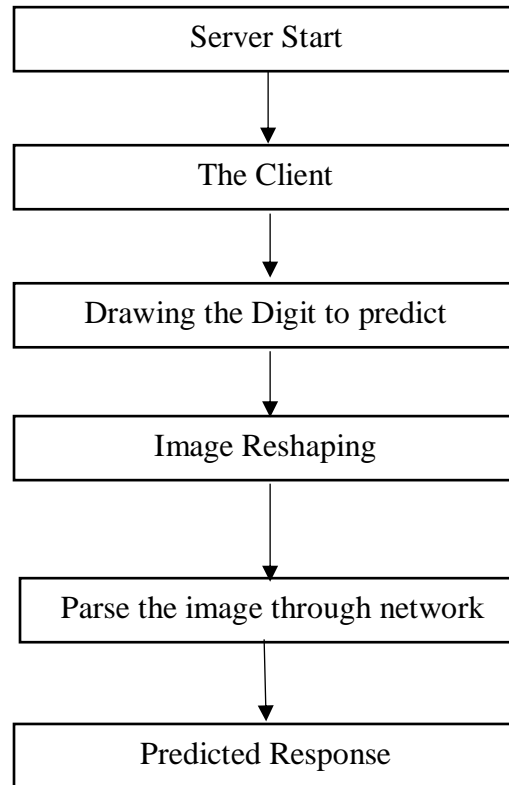
Incremental Model



### **GANTT CHART**

<b>Sr. No</b>	<b>Phase</b>	<b>Duration (Days)</b>	<b>Start Date</b>	<b>Finish Date</b>	<b>Actual Finish Date</b>
<b>1</b>	<b>Planning Phase</b>	<b>30</b>	5/2/2018	5/3/2018	5/3/2018
1.1	Defining Problem	10	5/2/2018	15/2/2018	17/2/2018
1.2	Confirm Project feasibility	10	18/2/2018	28/2/2018	28/2/2018
1.3	Project schedule	5	1/3/2018	5/3/2018	5/3/2018
1.4	Staffing project	2	6/3/2018	8/3/2018	8/3/2018
1.5	Launching project	1	9/3/2018	9/3/2018	9/3/2018
<b>2</b>	<b>Analysis Phase</b>	<b>25</b>	10/3/2018	4/4/2018	4/4/2018
2.1	Gathering information	10	10/3/2018	20/3/2018	20/3/2018
2.2	Defining system requirements	8	20/3/2018	28/3/2018	28/3/2018
2.3	Prioritize requirements	3	29/3/2018	31/3/2018	31/4/2018
2.4	Generate and evaluate alternatives	3	1/4/2018	3/4/2018	3/4/2018
2.5	Review the Documentation	2	4/4/2018	6/4/2018	6/4/2018
<b>3</b>	<b>Implementing Model</b>	<b>30</b>	7/4/2018	7/5/2018	7/5/2018
3.1	Construct the software components	23	7/4/2018	30/4/201	1/5/2018
3.2	Verify and test the components	4	1/5/2018	5/5/2018	5/5/2018
3.3	Training	1	6/5/2018	6/5/2018	6/5/2018
3.4	Documentation	2	7/5/2018	8/5/2018	9/5/2018
<b>4</b>	<b>Designing</b>	<b>11</b>	10/5/2018	21/05/2018	21/05/2018
4.1	Design user interface	5	10/5/2018	15/5/2018	15/5/2018
4.2	Design application architecture	4	16/5/2018	19/5/2018	19/5/2018
4.3	Design and integrate system controls	2	19/5/2018	21/5/2018	21/5/2018
<b>5</b>	<b>Support</b>		<b>22/5/2018</b>	<b>Till Date</b>	

### **DIGIT RECOGNIZER WORKFLOW**



### **CONTEXT DIAGRAM**

The Context Diagram shows the system under consideration as a single high-level process and then shows the relationship that the system has with other external entities (systems, organizational groups, external data stores, etc.). Another name for a Context Diagram is a Context-Level Data-Flow Diagram or a Level-0 Data Flow Diagram. Since a Context Diagram is a specialized version of Data-Flow Diagram, understanding a bit about Data-Flow Diagrams can be helpful. Context Diagrams and Data-Flow Diagrams were created for systems analysis and design. But like many analysis tools they have been leveraged for other purposes. For example, they can also be leveraged to capture and communicate the interactions and flow of data between business processes. So, they don't have to be restricted to systems analysis.





A Context Diagram provides no information about the timing, sequencing, or synchronization of processes such as which processes occur in sequence or in parallel. Therefore it should not be confused with a flowchart or process flow which can show these things.

Some of the benefits of a Context Diagram are: -

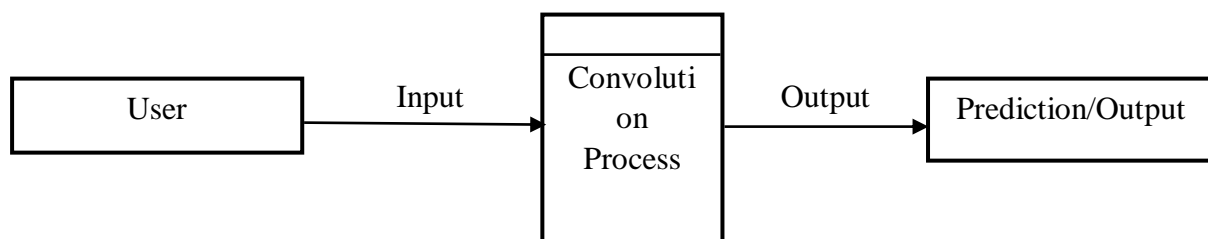
- Shows the scope and boundaries of a system at a glance including the other systems that interface with it.
- No technical knowledge is assumed or required to understand the diagram.
- Easy to draw and amend due to its limited notation.
- Easy to expand by adding different levels of DFDs.
- Can benefit a wide audience including stakeholders, business analyst, data analysts, and developers.

# DIGIT RECOGNIZER

## ELEMENTS OF CONTEXT DIAGRAM

	A square is an Environmental Entity (EE)—a source of, or destination for, data outside the system.
	An arrow is a data flow. Each data must have a unique identifier.
	A process bubble is a process that changes data. Process bubbles should be numbered at the top. The Process Label should be in a verb – object format. The bottom section identifies an actor or system component.
	An open rectangle is Data Store.

### Level 0



### **DATA FLOW DIAGRAM**

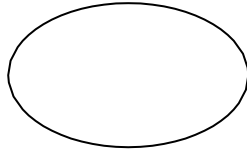
A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an Information System. A data flow diagram can also be used for the visualization of Data Processing. It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. This context-level DFD is then "exploded" to show more detail of the system being modeled.

A DFD represents flow of data through a system. Data flow diagrams are commonly used during problem analysis. It views a system as a function that transforms the input into desired output. A DFD shows movement of data through the different transformations or processes in the system.

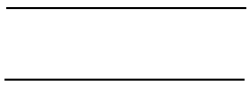
Dataflow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to restock how any system is developed can be determined through a dataflow diagram. The appropriate register saved in database and maintained by appropriate authorities.

## Data Flow Diagram Notation

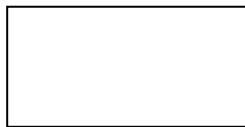
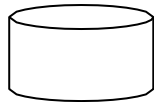
.



Function



File/Database

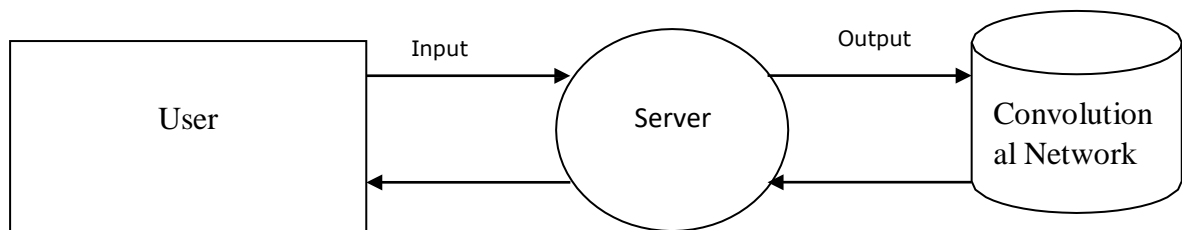


Input/output



Flow



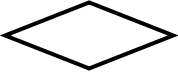
## Level 0



### **ENTITY RELATIONSHIP DIAGRAM**

A graphical model of the data needed by a system, including things about which information is stored & the relationships among them, produced in the structured analysis & information engineering. ER Diagram represents entities or tables and their relationships with one another.

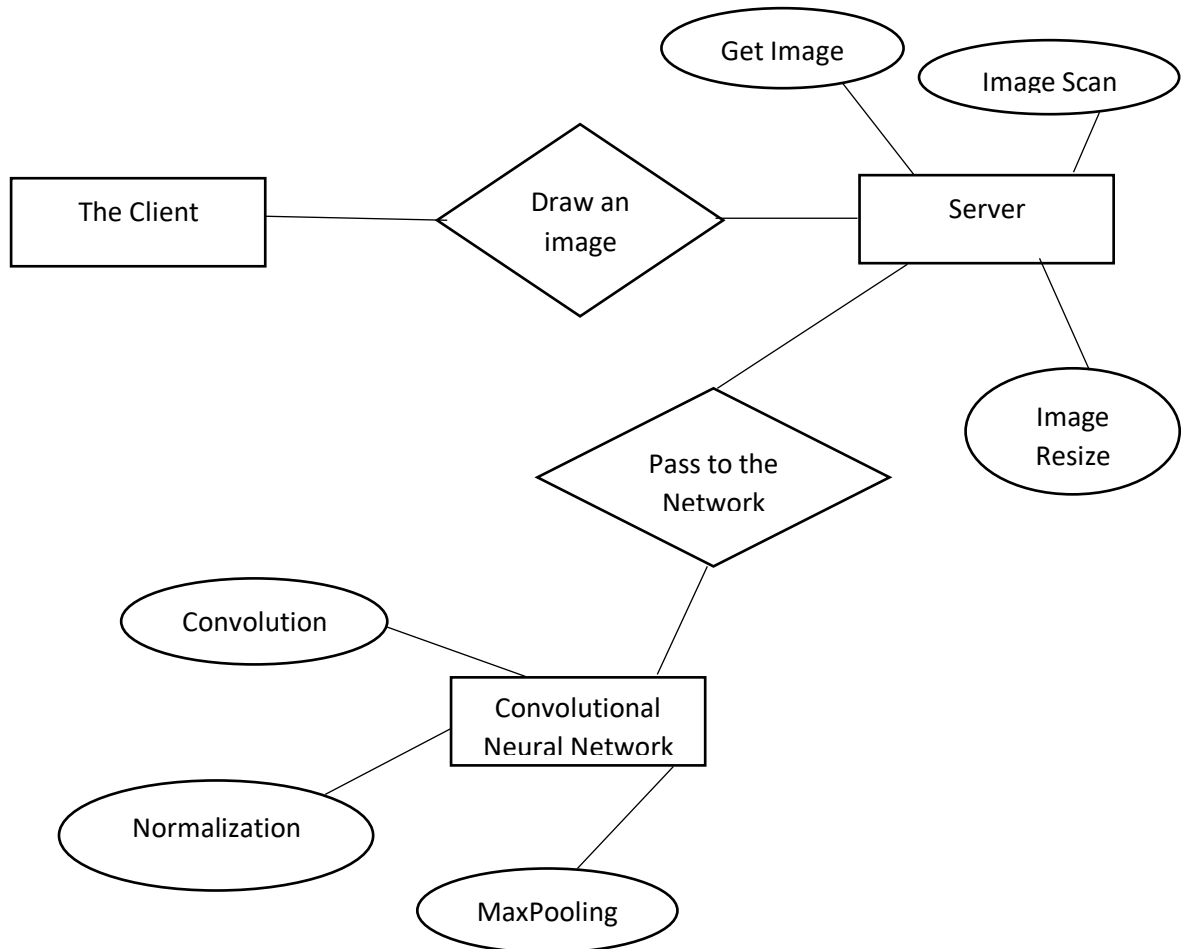
#### **Symbols:**

<b>Symbols</b>	<b>Represents</b>	<b>Description</b>
	Entity	An entity is a single object about which data can be stored. It is the "subject" of a table.
	Attribute	Attributes are the data that we collect data about an entity.
	Relationship	Relationships provide the structure needed to draw information from multiple entities.

Connection between components. An arrow generally indicates a flow from one component to another.



## ENTITY RELATIONSHIP DIAGRAM



## USE CASE

Use case diagram describes the behaviour of the system from user's point of view. It provides functional description of system and its major processes .it also provides graphic description is the user of a system and what kind of interaction of take place within the system.

### **1) Actor –**

An actor portrays any entity that performs certain roles in system. An actor in use case diagram interacts with the use case. It is shown outside the system hierarchy.

It is denoted by



### **2) Use case –**

A use case is a visual representation of a business functionality in a system .each use case is the sequence of transaction performed by the system. It is shown as ellipse in use case diagram.

It is denoted by

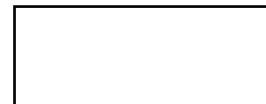


### **3) System Boundary –**

A system boundary defines the scope of what a system will be. A system cannot have infinite functionality. A system boundary defines the limits of the system.

The system boundary is shown as a rectangle spanning all the use case in the systems.

It is denoted by



### **4) Association –**

This is used to show the participation of actor in use case.

It is denoted by



### Relationships in Use Case –

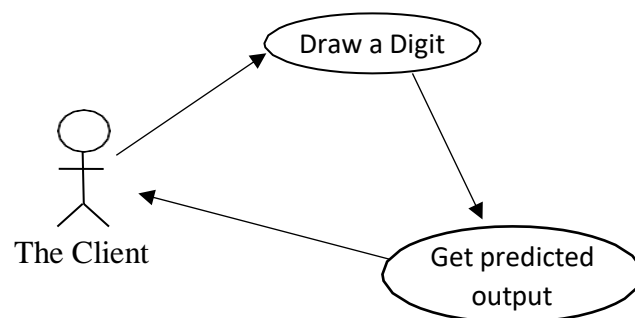
#### 1) Includes –

Include is used when 2 or more share some common portion in the flow of events. The stereotype <<include>> identifies relationship include. The arrowhead points towards child use case which is included in the parent use case.

#### 2) Extends –

In an extend relationship between two use cases. The child use case adds to the existing functionality a characteristics of the parent use case. It is represented as <<extend>>.

### USE CASE DIAGRAM



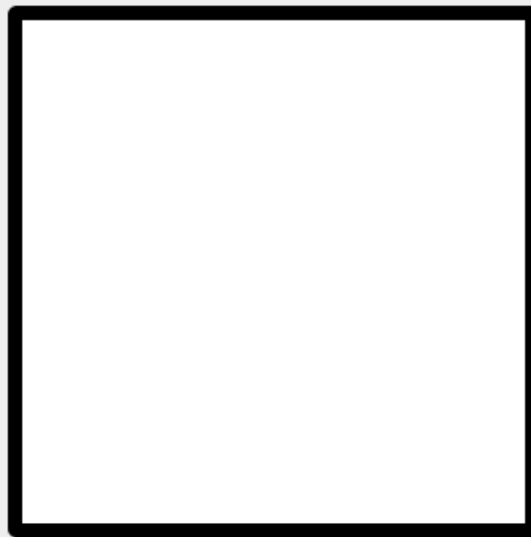
## Screen Shots

```
leonard@leonard: ~/pythonProject
leonard@leonard:~$ cd pythonProject/
leonard@leonard:~/pythonProject$ source MlpEnv/bin/activate
(MlpEnv) leonard@leonard:~/pythonProject$
```

```
leonard@leonard: ~/pythonProject/Alphanumeric
leonard@leonard:~$ cd pythonProject/
leonard@leonard:~/pythonProject$ source MlpEnv/bin/activate
(MlpEnv) leonard@leonard:~/pythonProject$ cd Alphanumeric/
(MlpEnv) leonard@leonard:~/pythonProject/Alphanumeric$ python3 app.py
/home/leonard/pythonProject/MlpEnv/lib/python3.5/site-packages/h5py/__init__.py:
36: FutureWarning: Conversion of the second argument of issubdtype from `float`
to `np.floating` is deprecated. In future, it will be treated as `np.float64 ==
np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2018-05-29 11:54:07.478748: I tensorflow/core/platform/cpu_feature_guard.cc:140]
Your CPU supports instructions that this TensorFlow binary was not compiled to
use: AVX2 FMA
Loaded Model from disk
```

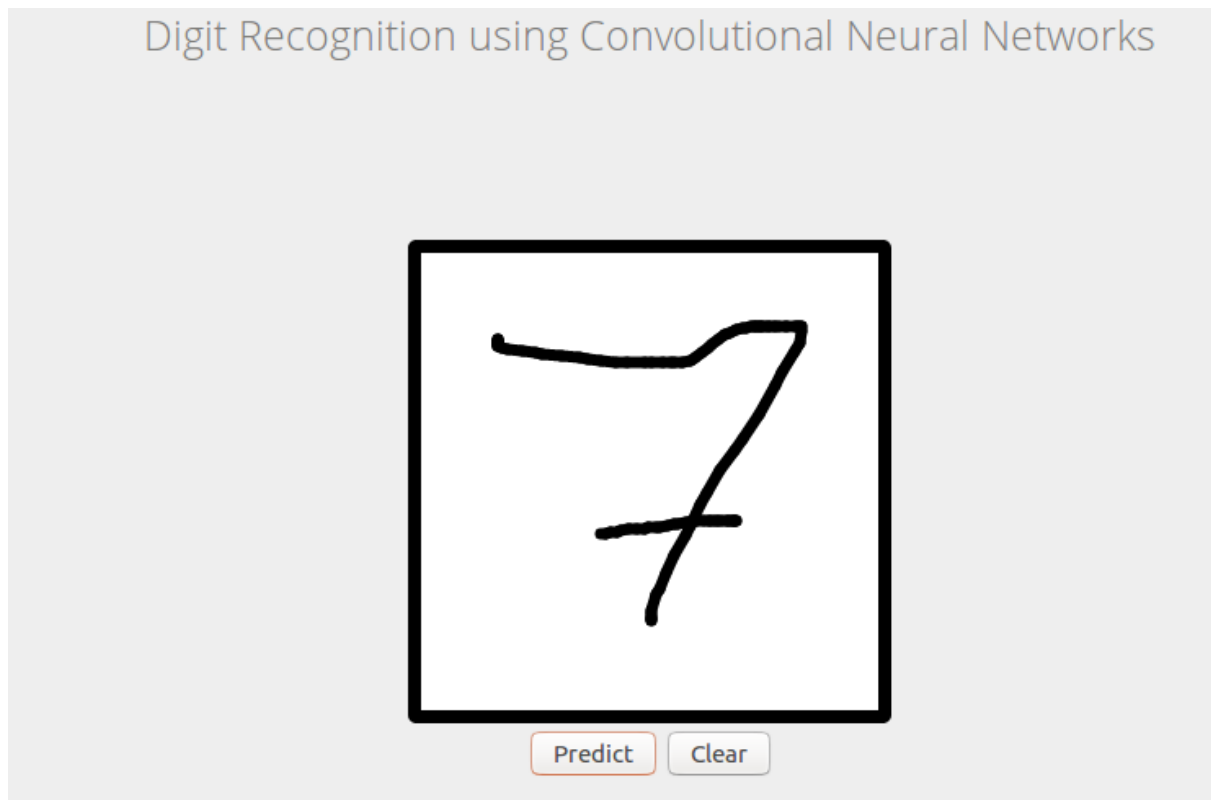
```
leonard@leonard: ~/pythonProject/Alphanumeric
leonard@leonard:~$ cd pythonProject/
leonard@leonard:~/pythonProject$ source MlpEnv/bin/activate
(MlpEnv) leonard@leonard:~/pythonProject$ cd Alphanumeric/
(MlpEnv) leonard@leonard:~/pythonProject/Alphanumeric$ python3 app.py
/home/leonard/pythonProject/MlpEnv/lib/python3.5/site-packages/h5py/__init__.py:
36: FutureWarning: Conversion of the second argument of issubdtype from `float`
to `np.floating` is deprecated. In future, it will be treated as `np.float64 ==
np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
2018-05-29 11:54:07.478748: I tensorflow/core/platform/cpu_feature_guard.cc:140]
Your CPU supports instructions that this TensorFlow binary was not compiled to
use: AVX2 FMA
Loaded Model from disk
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Digit Recognition using Convolutional Neural Networks



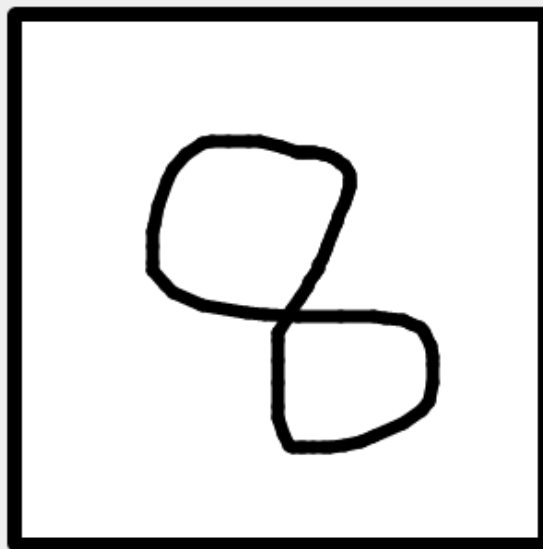
Predict

Clear



Predicted Output: [7]

### Digit Recognition using Convolutional Neural Networks



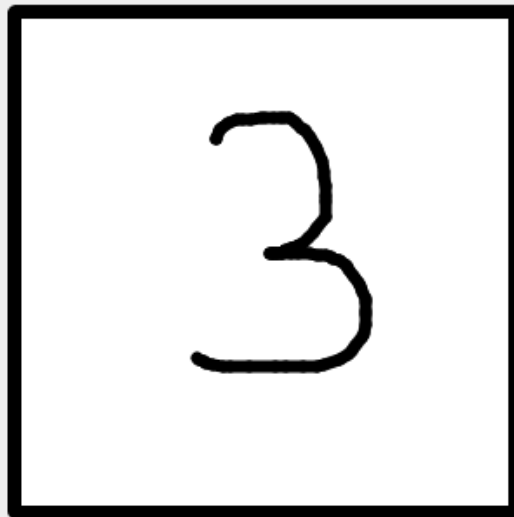
Predict

Clear

Predicted Output: [8]



### Digit Recognition using Convolutional Neural Networks



Predict

Clear

Predicted Output: [3]

### **TESTING**

Testing is the process of running a system with the intention of finding errors. Testing enhances the integrity of a system by detecting deviations in design and errors in the system. Testing aims at detecting error-prone areas. This helps in the prevention of errors in a system. Testing also adds value to the product by conforming to the user requirements.

The main purpose of testing is to detect errors and error-prone areas in a system. Testing must be thorough and well-planned. A partially tested system is as bad as an untested system. And the price of an untested and under-tested system is high.

The implementation is the final and important phase. It involves user-training, system testing in order to ensure successful running of the proposed system. The user tests the system and changes are made according to their needs. The testing involves the testing of the developed system using various kinds of data. While testing, errors are noted and correctness is the mode.

#### **Objectives of testing:**

The objectives of testing are: -

- Testing is a process of executing a program with the intent of finding errors.
- A Successful test case is one that uncovers an as-yet-undiscovered error.

System testing is a stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as per the user need, before the live operation commences. As stated before, testing is vital to the success of a system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. A series of tests are performed before the system is ready for the user acceptance test.

#### **Testing methods**

System testing is the stage of implementation. This is to check whether the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. The candidate system is subject to a variety of tests: on line response, volume, stress, recovery, security and usability tests. A series of tests are performed for the proposed system is ready for user acceptance testing.

## **Testing steps:**

### **I. Unit testing**

Unit testing focuses efforts on the smallest unit of software design. This is known as module testing. The modules are tested separately. The test is carried out during programming stage itself. In this step, each module is found to be working satisfactory as regards to the expected output from the module.

### **II. Integration testing**

Data can be lost across an interface. One module can have an adverse effect on another, sub functions, when combined, may not be linked in desired manner in major functions. Integration testing is a systematic approach for constructing the program structure, while at the same time conducting test to uncover errors associated within the interface. The objective is to take unit tested modules and builds program structure. All the modules are combined and tested as a whole.

### **III. Validation**

At the culmination of the integration testing, Software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test begin in validation testing.

Validation testing can be defined in many ways, but a simple definition is that the validation succeeds when the software functions in a manner that is expected by the customer. After validation test has been conducted, one of the three possible conditions exists.

- The function or performance characteristics confirm to specification and are accepted.
- A deviation from specification is uncovered and a deficiency lists is created.
- Proposed system under consideration has been tested by using validation test and found to be working satisfactory.

### **IV. Output testing**

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in a specific format. The output format on the screen is found to be correct. The format was designed in the system design time according to the user needs. For the hard copy also; the output comes as per the specified requirements by the user. Hence output testing did not result in any correction for the system.

### **V. User acceptance testing**

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes whenever required.

This is done in regard to the following point:

- Input Screen Design
- Output Screen Design
- Format of reports and other outputs

### **VI. Loop testing**

Loops are the corner stone for the vast majority of all algorithms, implemented in software. And yet we often pay them little need while conducting software tests.

Loop testing is a white box testing technique that focuses exclusively on the validity of the loop constructs. Four different classes of loops can be defined: --

- Simple loops
- Concatenated loops
- Nested loops
- Unstructured loops

**Simple loops:** - The following sets of tests were applied to simple loops. Where  $n$  is the maximum number of allowable passes through to the loop.

- Skip the loop entirely.
- Only one pass through the loop
- $M$  passes through the loop where  $m < n$ .
- $N-1$ ,  $N+1$  passes through the loop.

**Nested loops:** - The following tests were applied as nested loops.

- Start at the innermost loop. Set all the others loop to minimum value.
- Conduct simple loop tests for the innermost loop while holding outer loops at there their minimum iteration parameter values.
- Work outward, conducting tests for the next loop, but keeping all other outer loops at minimum values and other nested loops to 'typical' values.
- Continue until all loops have been tested.

**Concatenated loops:** - Concatenated loops were tested using the same approach as mentioned for simple loops if each of the loops was independent of other.

Otherwise in case of non-independent loops the approach mentioned for nested loops was used.

**Creating test data:** - Though some test data were created at the time of testing individual modules, but this may not be sufficient for further testing of the system as a whole. So, new test data were created for all possible real life situations, which were though might occur.

**Test Case Design:** - Test case design focuses on a set of techniques for the creation of test cases that meet overall testing objectives. In test case design phase, the engineer creates a series of test cases that are intended to "demolish" the software that has been built.

Any Software Product can be tested in one of the two ways:

Knowing the specific function that a product has been designed to perform, tests can be conducted that demonstrates each function is fully operational, at the same time searching for errors in each function. This approach is known as Black Box testing.

Knowing the internal working of a product, tests can be conducted to ensure that internal operation performs according to specification and all internal components have been adequately exercised. This approach is known as White-Box testing.

Black box testing is designed to uncover errors. They are used to demonstrate that the Software functions are operational; that input is properly accepted and output is correctly produced; and that integrity of external information is maintained (e.g. Data Files). A black box examines some fundamental aspects of a system with little regard for the internal logical structure of the software.

White box testing of Software is predicted on close examination of procedural details. Providing test cases that exercise specific sets of conditions and / or loops tests logical paths through the software. The “state of the program” may be examined if the expected or asserted status corresponds to the actual status.

A combination of white box testing and black box testing was applied in order to find as many errors as possible and debugging them and checking the working of the program.

A bottom-up approach was used during the testing as testing was started from the smallest unit or function and then integrating with others and in the end checking a testing the whole program together as a single entity.

The customer also applied a series of alpha tests at the developer’s site. The software was used in its natural setting with the developer looking over the shoulder of the user and recording errors and usage problem.

### **Security mechanisms**

The legitimate users are allowed to use the application. If the legitimate users share the authentication information then the system is open to outsiders.

### **Limitations**

- No mail facility.
- No message facility.
- No backup & restore facility.

## **Future scope and further enhancement**

This applications involves basic features of ‘Computer Vision’ but next update will provides more advance features of computer vision like text extraction from image, object detection using OpenCV. The future implementation will be helpful for building up a real-time computer vision application.

## **Conclusion**

The project entitled “Digit Recognizer” is developed using Python as front end and TensorFlow as the backend to compute and give predicted output to the client screen. This project covers only the basic two features of computer vision.

However these features are just start-up in building more complex computer vision app. Moreover extra features can be identified and incorporated in the future. Next update will be around six months with the features mentioned above.

In order to accommodate additional features it will take some time and effort to understand the requirement and converting it into a better application.

## **Reference**

<https://keras.io/why-use-keras/>

[https://www.tensorflow.org/versions/r1.1/get\\_started/mnist/beginners#the\\_mnist\\_data](https://www.tensorflow.org/versions/r1.1/get_started/mnist/beginners#the_mnist_data)

<https://stackoverflow.com/questions/35017160/how-to-use-virtualenv-with-python>

## **An introduction to Convolutional Neural Network**

Keiron O'Shea<sup>1</sup> and Ryan Nash<sup>2</sup>

<sup>1</sup> Department of Computer Science, Aberystwyth University, Ceredigion, SY23 3DB

[keo7@aber.ac.uk](mailto:keo7@aber.ac.uk)

<sup>2</sup> School of Computing and Communications, Lancaster University, Lancashire, LA1 4YW

[nashrd@live.lancs.ac.uk](mailto:nashrd@live.lancs.ac.uk)

**Abstract.** The field of machine learning has taken a dramatic twist in recent times, with the rise of the Artificial Neural Network (ANN). These biologically inspired computational models are able to far exceed the performance of previous forms of artificial intelligence in common machine learning tasks. One of the most impressive forms of ANN architecture is that of the Convolutional Neural Network (CNN). CNNs are primarily used to solve difficult image-driven pattern recognition tasks and with their precise yet simple architecture, offers a simplified method of getting started with ANNs.

This document provides a brief introduction to CNNs, discussing recently published papers and newly formed techniques in developing these brilliantly fantastic image recognition models. This introduction assumes you are familiar with the fundamentals of ANNs and machine learning.

**Keywords:** Pattern recognition, artificial neural networks, machine learning, image analysis.