# PyFlow

**Deep Learning Framework**

# Description

In this project we created a full deep learning framework to get the full experience of the deep learning engineer not only by using the well-known frameworks but also by developing our own framework , the framework contains Data module which help in loading the data and preprocessing it , NN module which contain layers as Dense Conv and pool , and also conation optimizer class , activation functions and Evaluation Metrics class , the framework contain utils module which help in saving and loading parameters and also contain visualization module which help in giving the full experience by plotting the results vs the training data .

**Product Perspective**

This project is a simple Deep learning frame work to help understanding the neural network concepts.

**General Capabilities**

The frame work is composed of several modules which will help building any efficient conventional NN model or CNN model.

**General Constrains**

The user build CNN model without wasting time on difficult interfaces and get the full experience.

**User Characteristics**

The user who uses this framework is interested in making Deep learning models easily and professionally.

**Environment Description**

We are using Colab notebook in our development to speed up the process using google servers.

# System Requirements:

## Functional requirements:
- Implement Data module
- Implement NN module to design different architectures
- implement visualization module
- implement utils module
- The modules must be combatable with each other

## Nonfunctional requirements:

### Product requirements:
The Model execution speed is fast as it executes in less than 1ms per example. It takes memory size of max 1GB. The system should have fast response time as it responds in 10ms after failure and it should tolerate common types of faults. Anyone can get it easy using pip install.
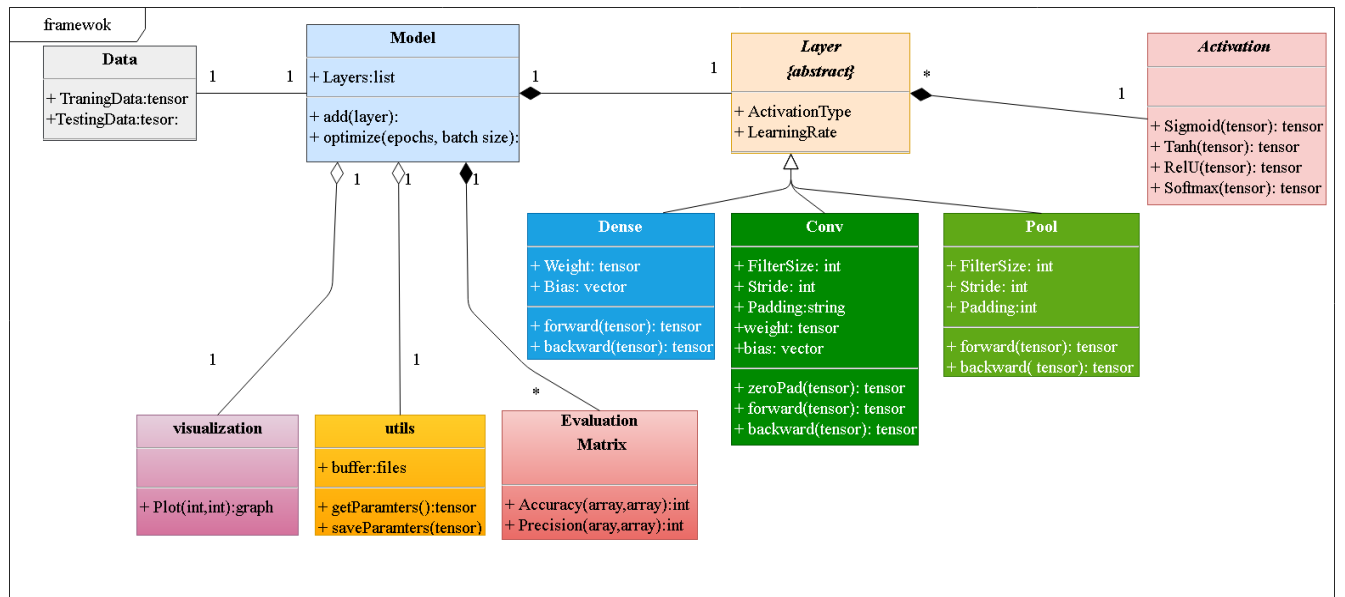
### Organizational requirements:
It is written in python using Colab notebook. The delivery time is on January 25st. It should conform to IEEE and ISO standards.

### External requirements:
The system should conform to all applicable local and international laws.

# Class Diagram:

**framewok**

### Data
+ TraningData:tensor
+TestingData:tesor:

### Model
+ Layers:list

+ add(layer):
+ optimize(epochs, batch size):

### Layer
### {abstract}
+ ActivationType
+ LearningRate

### Activation

+ Sigmoid(tensor): tensor
+ Tanh(tensor): tensor
+ RelU(tensor): tensor
+ Softmax(tensor): tensor

### Dense
+ Weight: tensor
+ Bias: vector

+ forward(tensor): tensor
+ backward(tensor): tensor

### Conv
+ FilterSize: int
+ Stride: int
+ Padding:string
+weight: tensor
+bias: vector

+ zeroPad(tensor): tensor
+ forward(tensor): tensor
+ backward(tensor): tensor

### Pool
+ FilterSize: int
+ Stride: int
+ Padding:int

+ forward(tensor): tensor
+ backward( tensor): tensor

### visualization

+ Plot(int,int):graph

### utils
+ buffer:files

+ getParamters():tensor
+ saveParamters(tensor)

### Evaluation Matrix

+ Accuracy(array,array):int
+ Precision(aray,array):int

1   1   1   1

1   1

1   1

*

# Sequence Diagram:

| Actor | Data | Model | Layer(Dense,Conv,Pool) | visualization | utils |
|-------|------|-------|------------------------|---------------|-------|

GetData()

Training and Testing data

**Repeat**

add(Layer)

Init()

return

return

Fit()

Loop on the training examples

Plot()

graph

Save()

return

Results