TEXT CLASSIFICATION

Project report submitted in partial fulfilment of the requirement for the degree of Bachelor of Technology in

Computer Science and Engineering/Information Technology

By

Pratyush Kumar (161202)

Varan Kaushal (161255)

Under the supervision of

Dr. Hari Singh

(Assistant Professor (Senior Grade), Dept. of CSE & IT.)

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat

Candidate's Declaration

We hereby declare that the work presented in this report entitled **" Text Classification "** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of our own work carried out over a period from July 2019 to December 2019 under the supervision of Dr. Hari Singh (Assistant Professor (Senior Grade), Dept. of CSE and IT).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Pratyush Kumar (161202)

Varan Kaushal (161255)

PRATYUSH KUMAR Digitally signed by PRATYUSH KUMAR Date: 2020.07.15 19:32:33 +05'30'

varank aushal Digitally signed by varankaushal Date: 2020.07.15 17:39:23 +05'30'

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Hari Singh Digitally signed by Hari Singh Date: 2020.07.23 09:42:36 +05'30'

Dr. Hari Singh

Assistant Professor (Senior Grade), Dept. of CSE and IT

ACKNOWLEDGEMENT

We take this opportunity to express our profound gratitude and deep regards to our guide Dr. Hari Singh **(** Assistant Professor (Senior Grade), Dept. of CSE and IT) for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him, time to time shall carry us a long way in the journey of life on which we are about to embark.

We are also obliged to staff members of JUIT College, for the valuable information provided by them in their respective fields. We are grateful for their cooperation during the period of our assignment.

Lastly, we thank almighty, our parents and our classmates for their constant encouragement without which this assignment would not have been possible.

Contents

List of Figures & Tables

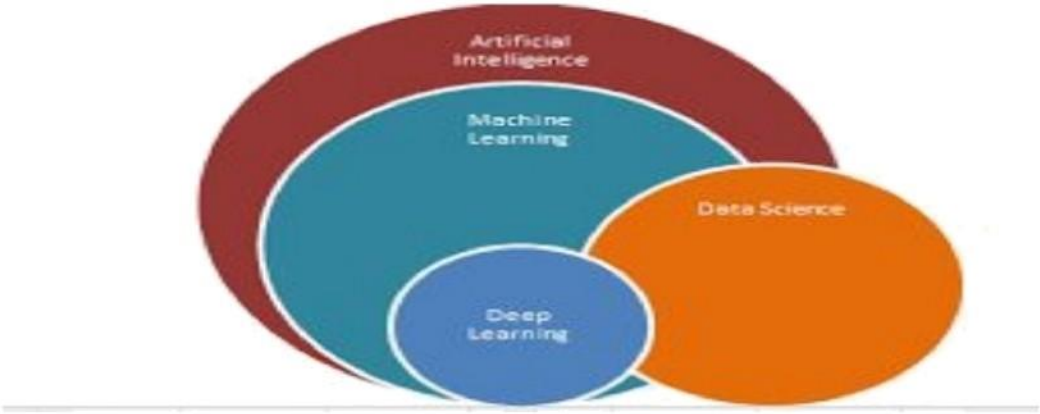
I

1. Introduction

1.1 Abstract

Since the invention of the first artificial neuron, McCulloch Pitts Neuron by Warren McCulloch and Walter Pitts in 1943 which works on dassifying two seg ts using a linearly separable line, the world has witnessed evolutionary development in the field of Artificial Intelligence and its subdomains Machine Learning and Deep Learning.

1.1.1 Corr€?lation betweenArtificial intelligenc€§, Machine L€§arning, Deep Learning and DataScience



Source: http s://miro.medium.com/max/922/1*32k RO9NhW pTq 1210-cZrpA.png

Figure 1.1) Correlation between AI, ML, DL and data science.

From the above figure (Vein Diagram) we could understand that deep learning is the subset of machine learning which in turn is the subset of artificial intelligence and the data science overlaps with some aspects of all three as it also involves aspects such as data collection (which can be either done manually or through sensors) and basic visualisation and analysis (calculating mean/ median/ etc.) and thus some aspects of it lies outside the spectrum of AI, ML, and DL.

Deep Learning Models has accelerated the pace of development and deployment of machine learning applications in our daily lives from sequence generation to image classification by leveraging the advantage of being highly efficient to be trained on large size of dataset and is used primarily in achieving all machine learning task.

Our endeavour in this project is to gain leverage of the deep learning models developed over the past years to classify text characters of two Indian languages i.e. Hindi and Tamil.

Text Classification has become an important application of Deep Learning in today's world and one of the world's most intense research areas. If a mobile application could classify text of different languages correctly, then it could be efficient for common users to communicate effectively anywhere and at anytime.

In case of language translation i.e translating any foreign language to user's own native language, it must go@rough the first step of identifying the text correctly, which is an important application of image processing and subsequently computer vision today.

1.1.2 What is Computer Vision and Image Processing ?

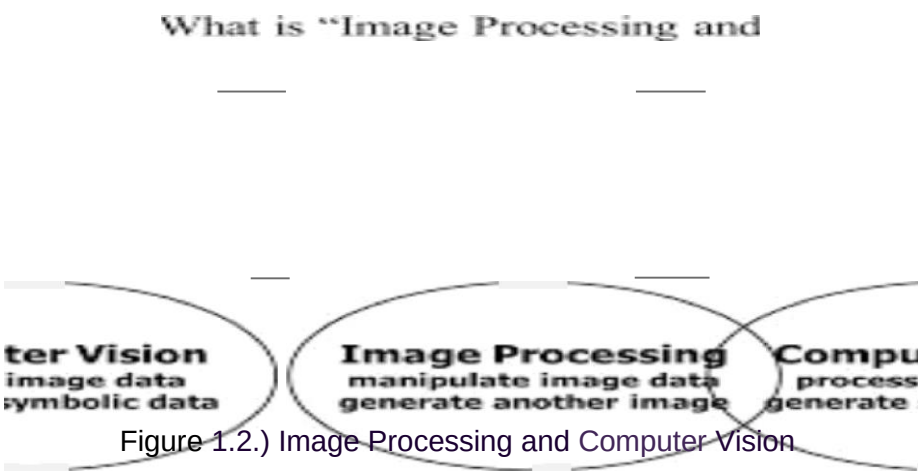


Figure 1.2.) Image Processing and Computer Vision

Image Processing: It is a technique which takes an image as an input and generates an image as output by performing some modification on the desired attributes (color/ intensity/ use removal/ compression etc). Image processing eases the task of performing computer vision.

Computer Vision: It is an interdisciplinary science that allows a computer to gain a high-level understanding of digital images and videos and determine what the computer "sees" or "recognizes" objects and automate the task done by this ability on the various application which requires these functionalities.

Object Detection for self-autonomous cars, the model building of a location, image classification on social media applications, providing vicarious experience through augmented reality (AR) or virtual reality (VR) are some of the commonly observed applications of computer vision.

1.2 Probl€?m Stat€?FD€?Flt:

We have collected data of two indian languages (Hindi and Tamil) of all the possible characters and our task is to train a deep learning model so that it could successfully classify the characters of the dataset curated and pre-processed (multi-class classification) with maximum attainable accuracy by trying it the model on various possible combinations of hyperparameters (epoch/ learning rate/ optimisation algorithm/ activation functions) and minimising the error using backpropagation technique so that it can make more no. of correct predictions both on the training data and test data.

Our project is an application of supervised learning in machine learning.

Supervised Learning : Given a labeled training dataset , we train it on some model of our own by inferring the relationship between inputs(x's) and output(y's) and try to find the best updated parameter and test the model on unseen data points.

Unsupervised Learning : Given an unlabeled dataset , we try to either group the data points on the basis of some attributes(demographics/behaviour/like/dislike) or generate new sequence of inputs based on the available inputs.

"Supervised Learning(Classification/Regression) has created 99% of all economic value in Artificial Intelligence" Andrew NG (CEO , deepIearning.ai)

1.3 Our Approach

We have approached this project through the well known six elements of machine learning, used in learning machine learning.

There are six elements of machine learning

1. Data
2. Defining a Task
3. Applying Model
4. Calculating Loss
5. Learning Algorithm
6. Evaluation

1. Data : (The fossil fuel of machine learning)
   A typical dataset required to perform any ML prediction is of high dimensions meaning it consists of millions of rows/data points/observations with typically thousands or may be millions of columns/parameters/features.A dataset could be presented with inputs as well as its corresponding outputs which is ideal for performing any supervised learning task by learning the relationship between ilp and o/p and if the dataset doesn't contain any corresponding output w.r.t to inputs then we can only perform unsupervised learning task.

   Our source of the data for this project of collecting hindi and tamil language is Kaggle (an online competition website).

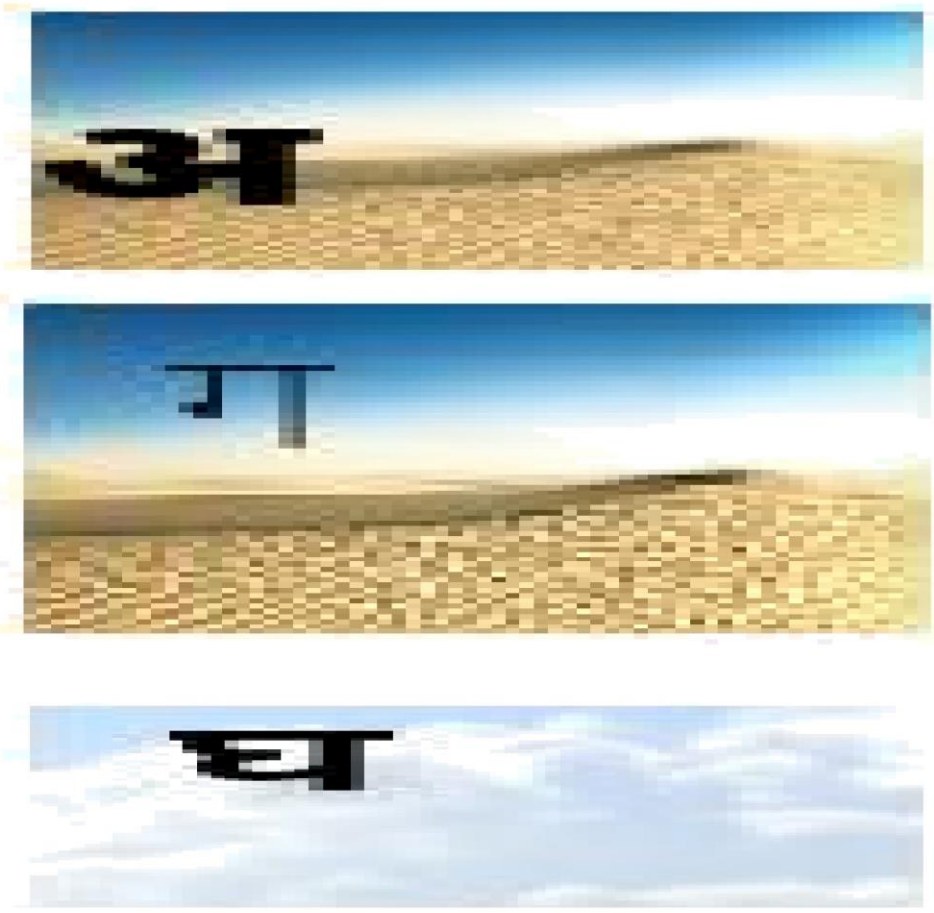


Figure 1.3) Characters of Hindi Language

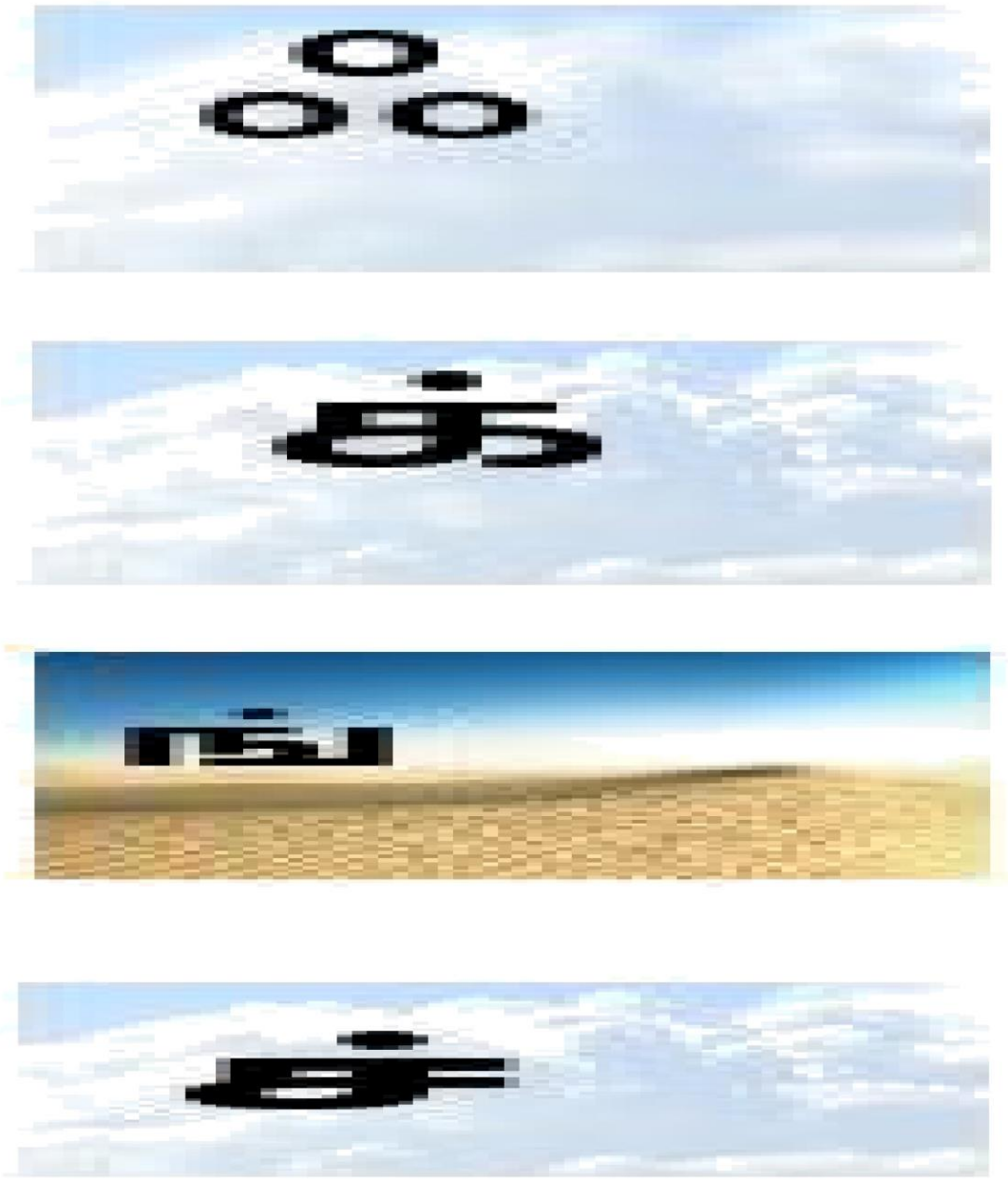


Figure 1.4) Characters of Tamil Langauage

Data Curation (Pre-processing the data) :

The data that has to be feeded to the model for training should be in machine readable form i.e. it should be encoded as number.

In the above case image could be represented in 3 channels, i.e RGB and could be represented in pixel value.

2. **Task:** (**Setting an objective of the project with the curated dataset**)

Based on the procured/curated dataset we can define our task accordingly. If we have labeled training dataset i.e. it contains input(x's) and its corresponding labels(y's) we can easily perform supervised learning(classification/regression) and if we don't have labeled training dataset doesn't contains corresponding labels(y's) we can only perform unsupervised learning(clustering/generation).

Classification(binary/multi) : The data point is tried to be mapped with some category/label. The label could be binary such as (like/dislike) or multi such as classifying the alphabets of english language(A-Z).

We are performing Multi-class classification in this project.

3. **Model** : (**Mathematical formulation of a task**)

y=f(x)[unknown relation]

Our approximation function : ^y=^f(x)

A ML/DL model is typically the approximation of an ml engineer for finding the relationship between data points(x's) and its corresponding labels(y's) denoted by y=f(x). The model could be simple or complex depending on the no. of parameters it has.

In this project we have worked on Convolutional Neural Network model and it's various architectures proposed over the years by the researchers.

4. Loss Function : (How could we say which model best fit for estimating the correct relationship for the task ?)

e
Loss function calculates the difference between true output (y) and the approximated value (^f(x)) . It is represented with L.

If L=0 then the estimate of our model is exactly accurate

Different loss functions are '

a. Square Error Loss
b. Cross Entropy Loss
c. KL divergence
d. Hinge Loss
e. Huber Loss etc

The lower the loss better the model is in terms of accurate prediction.

5. Learning Algorithm : **(How do we estimate different parameters ?)**

Under this element, the success of machine learning lies.

Parameter estimation in machine learning is a kind of a search operation.We can compute the parameters through learning algorithm and it becomes an optimisation problem where we try to optimise the parameters by minimising the loss. Hence, learning algorithm and loss function goes hand in hand.

Some of the popular learning algorithm/optimisation solver are

a. Gradient Descent
b. Adagrad
c. RMSProp
d. ADAM
e. Backpropagation

6. Evaluation : (How do we compute the accuracy of a MLJ'DL model ?)

Accuracy = (No. of correct prediction)/(Total no. of prediction)

Calculating accuracy indicates how efficient the model is and is more interpretable for the end user than the loss function.

Top-K Accuracy : Out of the top-k prediction made by the prediction if we can find the correct output among these then we can accept the model.

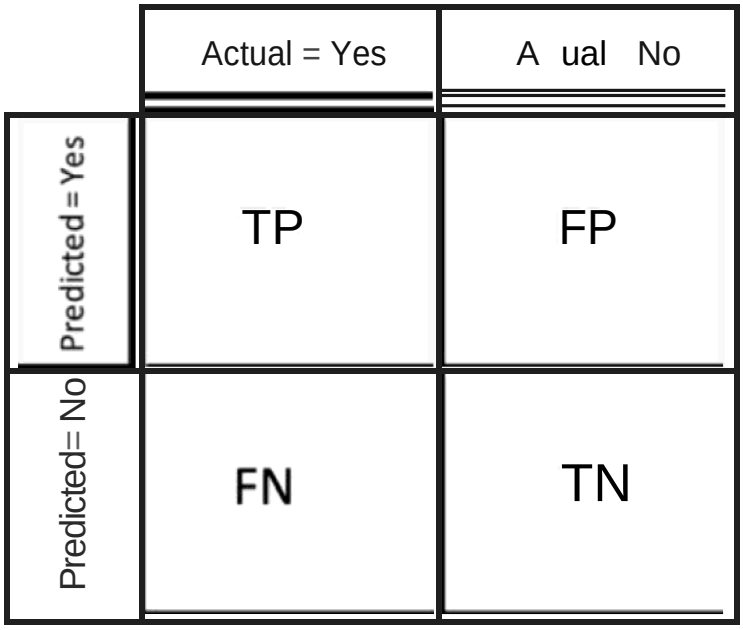
Top-K Accuracy = (No. of correct prediction made in top-k) / (Total no. of prediction)

Evaluation has to be done on the test data (data points which hasn't been seen by the machine).

The standard evaluation metric in case of object detection where some action is required to be taken are precision and recall.

Precision = (No. of correct action) / (Total no. of action taken)

Recall = (Actual no. of correct action) / (Total No. of times correct action to be taken)



Source : https://miro.medium.com/max/848/1”7SgzmX05T81Ojaor9s5HWO.png

Figure 1.5) Representing correct prediction and wrong prediction using Confusion Matrix

Example: Out of 1200 total images, if the model could correctly classify 1080 images then the model is said to be trained with the 90 %age accuracy.

2. Literature Survey

2.1. List of Literatur€§ **Survey§2d**

We have surveyed research paper written by well known researchers over the past years, over the work they have done in the field of Image Classification from the improved Yuan Lecun (improved LeNet 5) original research paper of 1998 to the most efficient and better than human performance of the model ResNet developed by Microsoft in 2015.

A.) Improved Lenet-5

B.) AlexNet

C.) VGG-Net

D.) GoogleNet

E.) Resnet

2.2 Description of LitE§ratur€§ SurvE§yed

A.) " **Digital** Recognition based on **improved** LeNet Convolutional Neural Network " by Li Sun, Lishan Jia

In this paper we have studied,

An improved model over the LeNet-5 model published in 1998 which contains 8 layers including i/p and o/p layers in contrast to original network consisting of 9 layers.

The activation function used is logistic compared to hyberbolic tangent function used in original network and works best in the combination of learning rate=1, epochs and batch Size is 25.

Some of the pertinent improvements made in the paper were

Parameters (Original Network of LeNet 5) = 60, 000

Parameters (Improved LeNet-5) = 32, 194

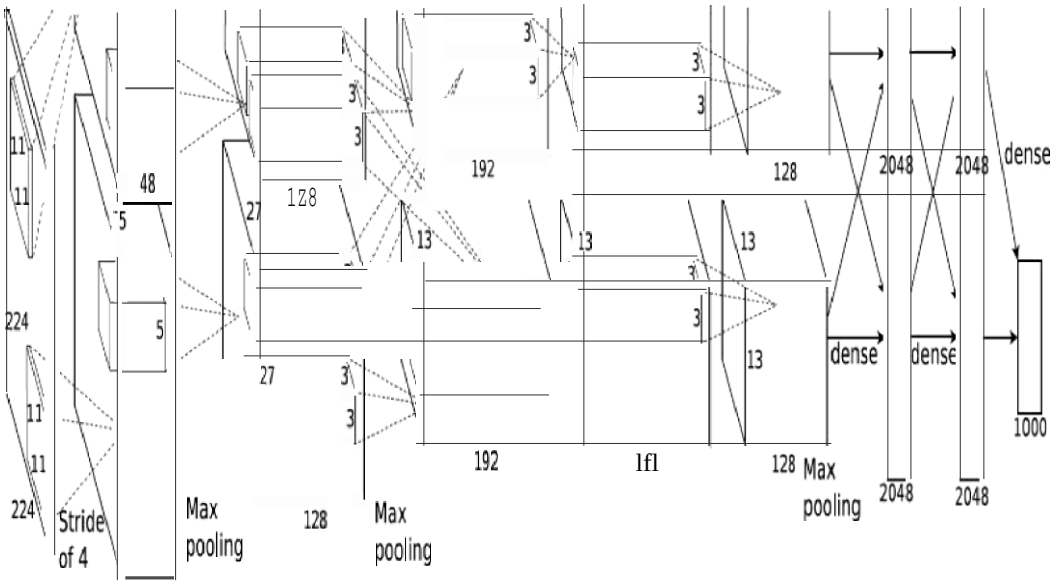
Misrecognition Rate (Original Network of LeNet 5) = 0.98%

Misrecognition Rate (Improved LeNet-5) = 0.86%

B.) “ALexNet" by Alex Krizhevsky, Geoffrey E. Hinton, Ilya Sutskever

In this paper delivered in 2012 at the ImageNet challenge, the authors of the paper presented a different kind of CNN Architecture which has been proven to be trained with the lowest possible error till 2012 of 16.4 %age.

The Cnn Architecture of AlexNet looks as follows:



Source: https://miro.medium.com/max/3072/1’qyc21qM0oxWEuRaj-XJKcw.png

Figure 2.1) Alexnet Architecture

The Top-5 error rate observed at the ImageNet dataset was 16.4 percentage.

3. " VGG-NET " by Karen Simonyan and Andrew Zisserman

This model gets popular by the name of VGG Net and is given in the year 2014 and emphasises on the addition of more and more depth to the model taking the depth of the model to 16 layers with the use of simple 3*3 filter.

The architecture of VGG-Net is as follow:

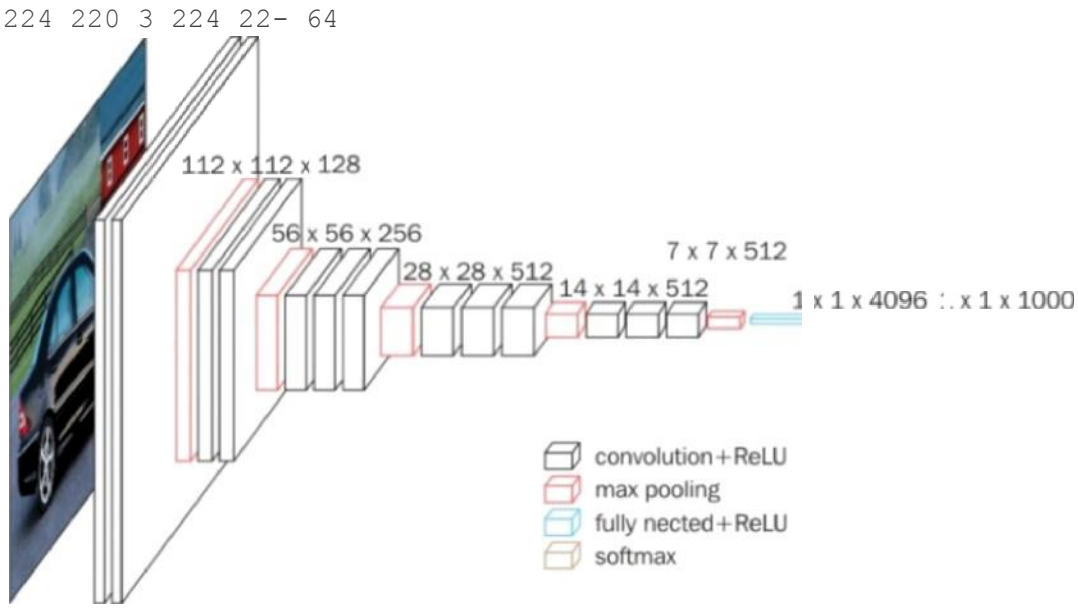


Figure 2.2.) VGG-Net Architecture

D.) Going Deeper with Convolutions (GoogleNet)

This model is even more deeper than AlexNet and VGG-16 as it takes the depth to even larger extent with introducing 22 layers into the picture. This model uses the concept of Inception model which signifies using multiple filters (1*1, 3’3, 5*5) and introduces the concept of auxiliary loss i.e checking the loss in between the layers to avoid the problem of vanishing gradient during backpropagation.

This model has been trained with the loss of 6.7 percentage.

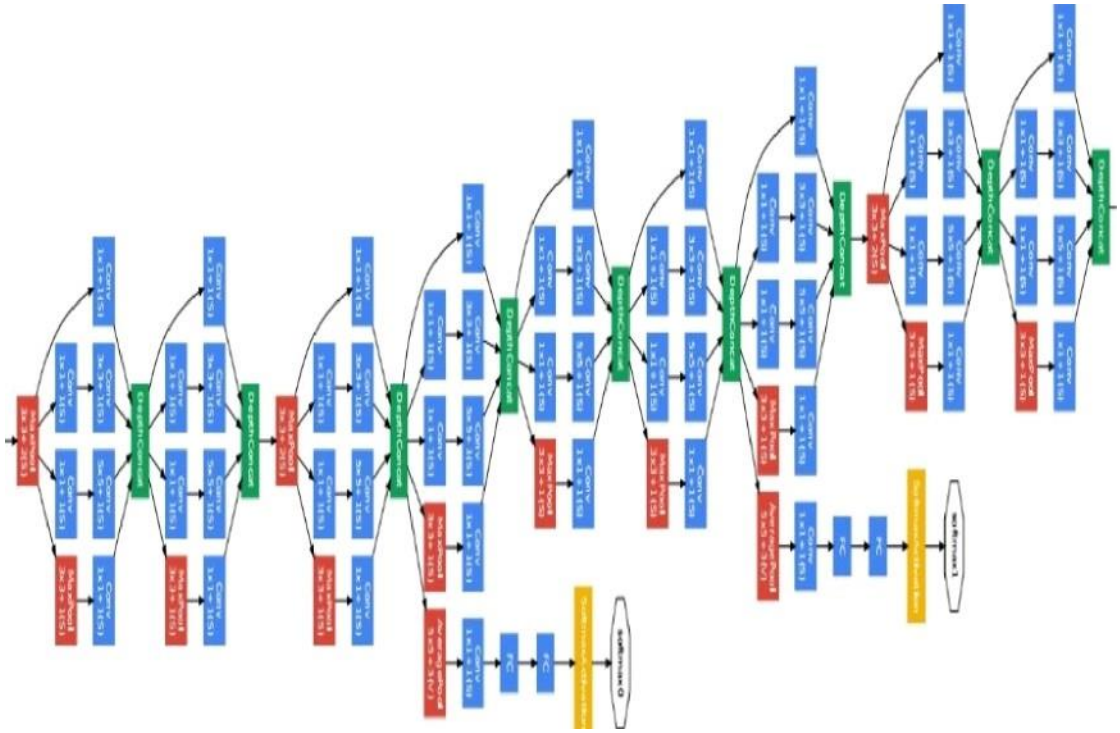


Figure 2.3) GoogleNet Architecture

E.) Deep Residual Learning for Image Recognition

This CNN architecture has three layers of architecture.

1.) Resnet 51
2.) Resnet 101
3.) Resnet151

This architecture introduces the concept of residual learning, i.e at every layer, the input is passed as it is and the residual of it too. Due to the depth it takes time to train the network. Resnet won the 101 architecture. The top-5 error rate was 3.57 percentage.

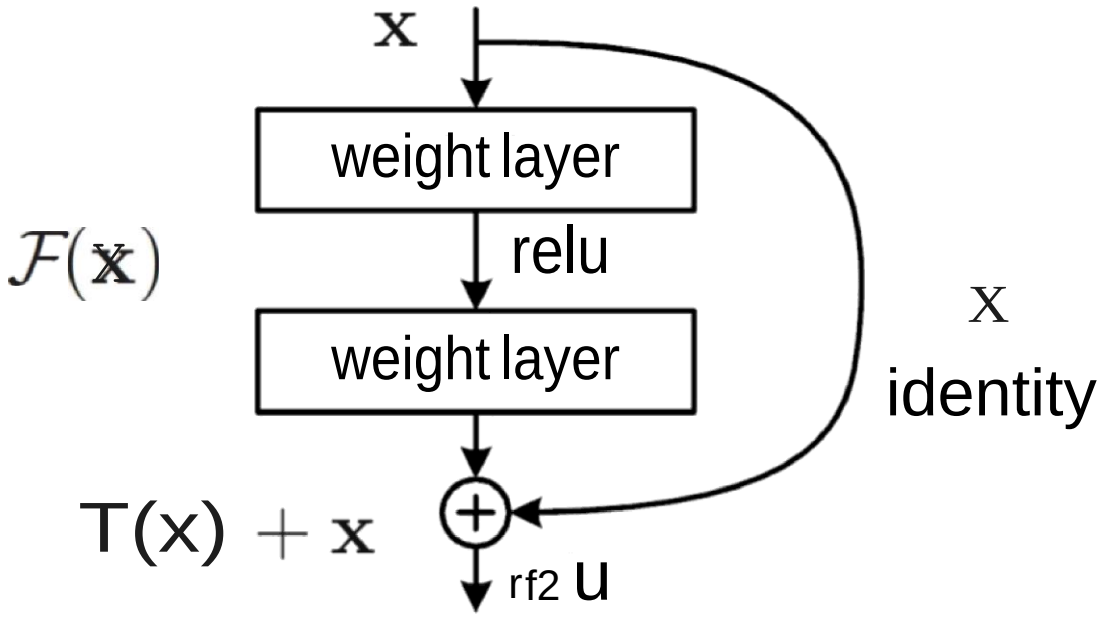


Figure 2.4.) Residual Learning

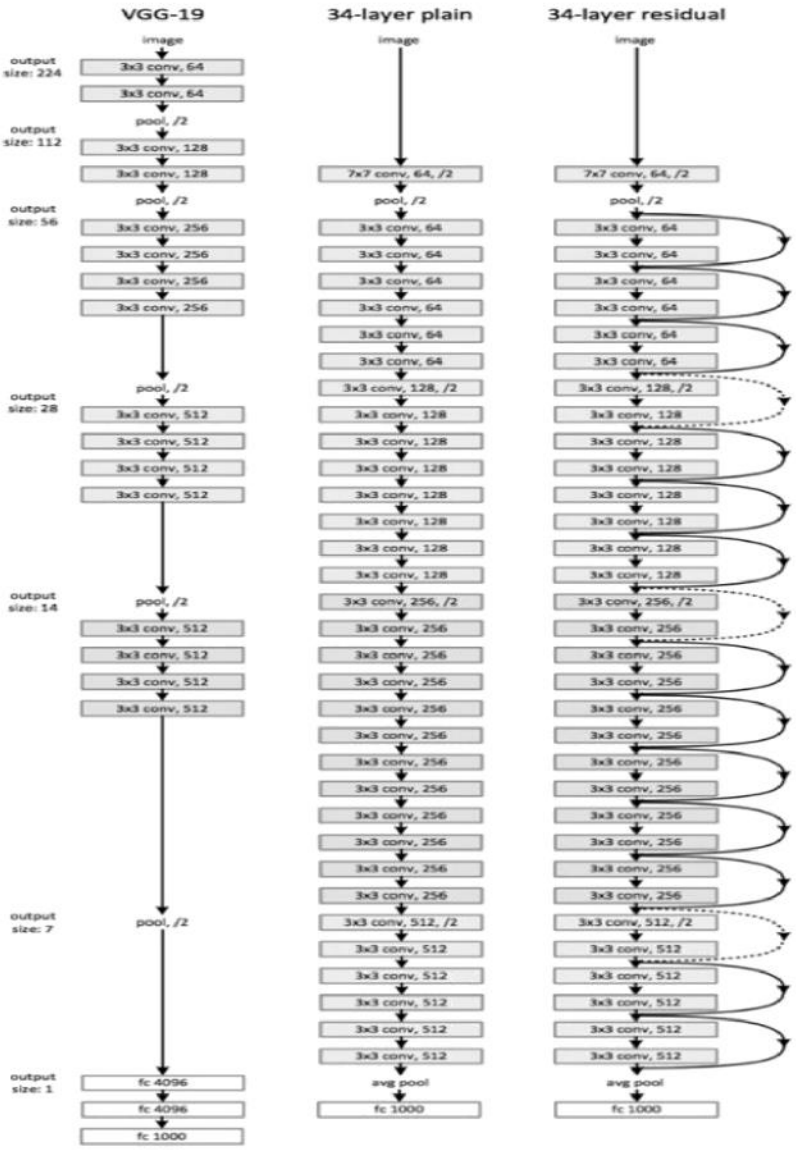


Figure 3. Example network architectures for ImageNet. **Left**: the VGG-19 model [41] (19.6 billion FLOPs) as a reference. **Middle**: a plain network with 34 parameter layers (3.6 billion FLOPs). **Right**: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. **Table 1** shows more details and other variants.

Figure 2.5) VGG-19 and Resnet architecture

3. System Development

3.1. Framework for training Deep Learning Models.

Some of the widely popular used framework for training deep learning model are:

A.) Tensorflow (developed by Google)

B.) Pytorch (developed by Facebook)

C.) Sonnet

D.) Keras

E.) MXNet

F.) Swift

G.) Chainer

H.) DL4J

I.) ONNX

3.2. Training D€§ep Learning Model using Pytorch FramE§wOFk

Originally developed by Facebook but used now a day by most of the companies and startups for training the model to achieve a specific task. Pytorch has some specific in-built function which can be used for minimising the training time of the model from hours to minutes specifically in the case of image classification.

Some of the in-built functions are:

A.) Compose (used in pre-processing the image)

B.) “NN” module helps in applying fully-connected layers and help in applying backpropagation and loss easily.

LOSSES in Pytorch:

Loss with PyTorch can be calculated a using nn module. PyTorch provides losses such as cross-entropy loss (nn. CrossEntropyLoss). In general, loss is assigned to the criterion. To actually compute the loss, you first define criterion then pass it in the output of your network and the correct labels. The output of the criterion is the loss for the network.

Autograd:

Torch provides a module called Autograd which calculates the gradients of tensors. We use Autograd to calculate the gradients of all our weight parameters with respect to the loss.

We can also download the model in pytorch with pretrained weights
Using the following code:

```
resnet = models.resnet18(pretrained= True)
```

The above is the example of downloading resnet architecture with pretrained weights.

The biggest advantage of pytorch over tensorflow is that it operates on a dynamically updated graph meaning we can change the architecture as per our wish and train it using various combination of hyperparameters.

Some of the common used pytorch libFaries art:

```
import torchvision.transforms as transforms
import torchvision.datasets as datasets
from torch.utils.data import DataLoader, Dataset, random_sp1it
import €orch.nn as nn
import €orcA.nn.functional as K

from torch.optim  mpcrt lr_sc&eduJer
import torchvision.models as models
```

Figure 3.1) Importing Pytorch Libraries

3.3 Using Cuda Proc€§ssor

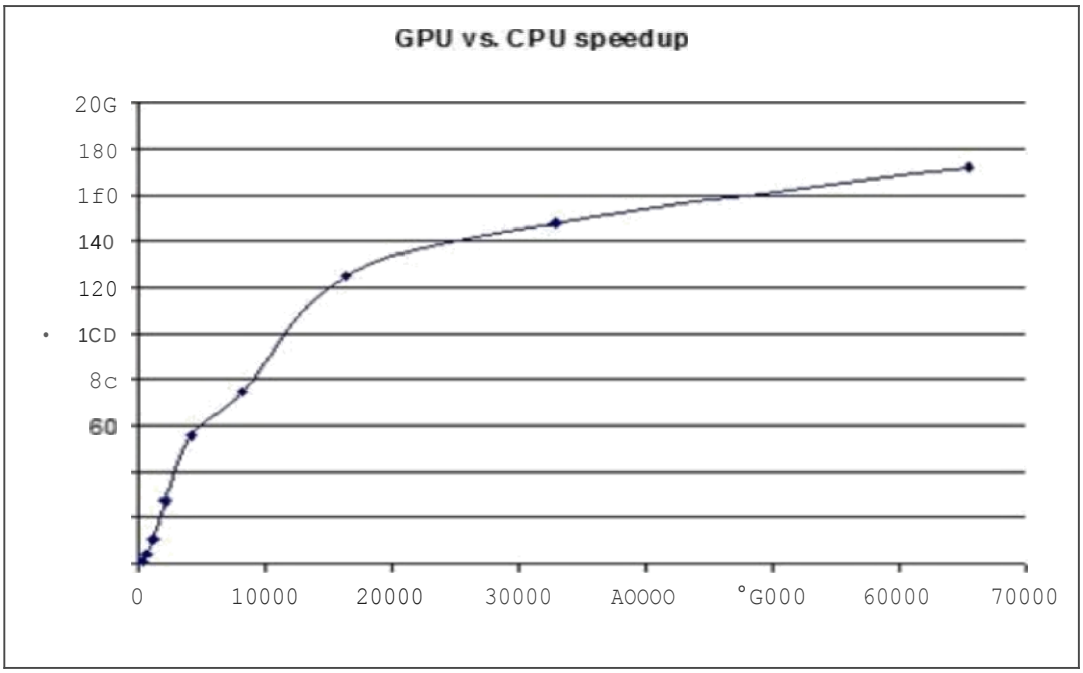


Figure 3.2.) GPU vs CPU performance

Matrix multiplication is a heavy task in training any deep neural network and takes a lot of time as with convolutional and fully connected layers, it may involve millions of parameters and generally training it on normal CPU could take up to much larger time depending on the dataset, but if we train it on GPU such as CUDA, the processing time decreases as the speedup observed is much higher in case of GPU.

3.4.) **Pri Proc$§sSiB§j thR Image**

The code for preprocessing the image is:

```
                                        transforms.Normalize([O i85, D.45C,
? 4D0], [0 229, 0 224, 0 225])])

train_loader   =   torch.utils.data.DataLoader(train_data,   batch_size=64,
shuffle=True)
test loader    =    torch.utils.data.DataLoader(test data.    batch size=b(,
shuffle-True)
```

Figure 3.3.) Preprocessing code

The above code converts the pixel values into tensors (a fancy name for arrays), and perform normalisation of the value.

Then after splitting the dataset, we can load the dataset into train loader and test loader as per our desired batch size of training and testing.

Note: We may have to apply different normalisation technique depending upon the model we intend to use to train the model.

3.5.) ImB§jR PFOCE§SSI B§j

In machine learning we deal with digital data of an image in order to make it suitable for processing , but an image may have different ranges of colour. so while scanning an image it generates analog data which is not suitable r extracting the right information from that image. To get the desired information first we need to convert the analog data of image to digital format. The process of gathering the required data from an image is referred as Image Processing. There are various mobile phone applications which help us to get desired photographs e.g. retrica. In-display fingerprint sensors also use the concept of image processing.

Sampling and quantization are the basics of image processing. We have various algorithms by which we can perform sampling and quantization. In sampling the coordinate values are digitized and in quantization the amplitude values are digitized.

Figure 3.4.) Illustration of Image Processing

Why Image Processing?

Image processing is not only limited to photoshop softwares, it has wide variety of applications in medical field, biometric verification, face and signature recognition, remote sensing, digital video processing and much more. In the field of machine learning, image processing is mostly used to proc s our raw data into useful form. Mostly, this is done to enhance the performance of machine learning algorithms by choosing the correct image processing technique.

The type of technique to be used depends on what our machine learning model aims at. The cost/loss functions in machine learning can help us determine that which technique fits good for best possible output.

In text classification models, the input data(training data) is processed before hand. It is a good deal to process the raw data with different techniques so that it is converted into useful format. It is a must that our model is trained only with fruitful information. This approach leads to improve the accuracy of our machine learning model.

Edge Detection

There are different techniques to process the image for gathering useful information, one of them is edge detection. As the name suggests this technique is used to detect edges in image by defining boundaries of an object present in image. The basics of edge detection is that it identifies the points on an image where the brightness levels changes sharply. By identifying the vertical and horizontal edges, we are able to define the boundaries of an object. There are many techniques for detecting edges. The detection techniques can be further classified on basis of first order and second order derivatives.

In first order derivatives the derivative of all the intensity values are taken and the point with highest derivative is considered. Examples of first order derivatives are: Sobel and Canny.In second order derivatives, differentiating the first derivative gives us the second derivative and finding the points for which it is zero gives us maximum or minimum. Example: laplacian.

Sobel

In sobel there are two types of masks for detecting edges. The vertical mask and the horizontal mask with size 3x3. ~~*nrfirnl mnrk in ii ñ In hntnrf rfiral h; nnd~~ horizontal mask is used to detect the horizontal eclges. Allotting more weights to them helps getting more edges.

Vertical mask:

-1	0	1
-2	0	2
-1	0	1

On applying vertical sobel filter on one of the text image that we have, we get the images with the vertical edges.

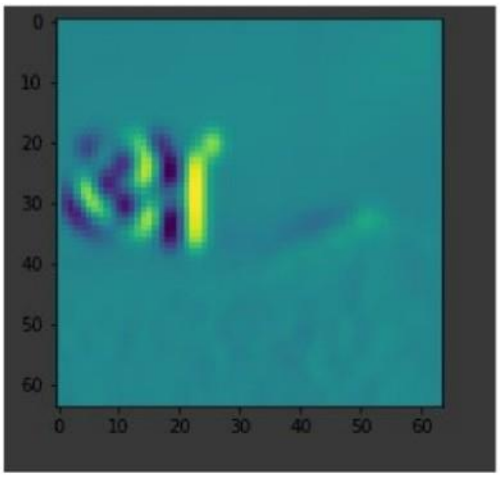


Figure 3.5.) Appling Vertical Sobel on the image

Horizontal mask:

-1	-2	-1
0	0	0
1	2	1

On applying horizontal sobel filter on one of the text image that we have, we get the images with the vertical edges.

Figure 3.6.) Applying Horizontal SObeI on the image

Canny

Canny edge detection is mostly used where we need smooth edges. It is mostly used in medical image analysis. There are five steps in Canny with which we get our desired output.

The image is smoothened with a Gaussian filter for noise reduction.

Equation for Gaussian Filter:

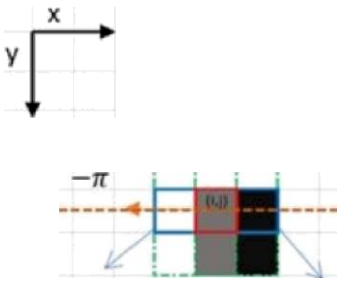
$$H, \quad \frac{\text{i}}{2.}\exp\left(-\frac{(\langle \quad (\text{k}+*))^2 \quad (\text{Q} \quad (\text{k}+\rangle))^2}{2}\right); 1 \leq i, j \leq (2k+1)$$

By using finite-difference approximations for partial derivatives we then calculate gradient magnitude(G) and orientation(6).

G and 0 are calculated as:

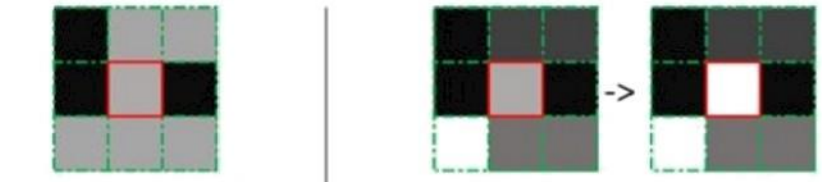
$$\theta(x, y) = arctan\left(\frac{\cdot y}{\cdot}\right)$$

We apply non-maxima suppression to the gradient magnitude for thinning the edges.



Then there is double threshold step in which we classify pixels as strong, weak and non-relevant.

Edge Tracking by hysteresis is the final step in which we process weak pixels to strong if one pixel around them is a strong.



Laplacian

It is an second order derivative edge detection techniqug which finds inner and outer edges rather than vertical and horizontal ones. Positive laplacian mask is used to find the outer edges of an image and negative mask is used to find the inner edges. To get the resultant image we don't have to use both the operators, either of them can give the resultant image by adding or subtracting the original image.

Positive Laplacian operator:

0	1	0
1	-5	1
0	1	0

Negative Laplacian operator:

0	-1	0
-1	5	-1
0	-1	0

On applying lapacian (gaussian) filter on the text image we get the edges in both the direction as;

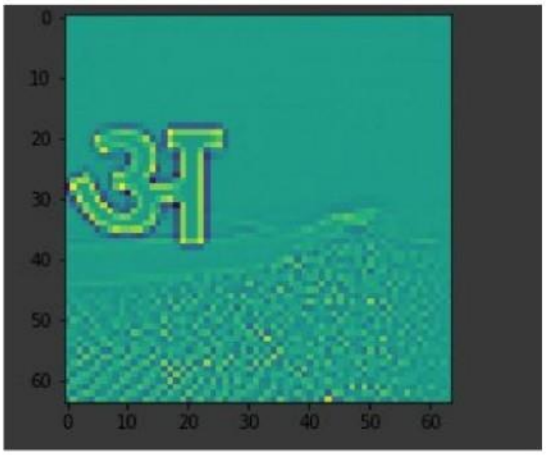


Figure 3.7.) Applying Laplacian filter on the image

3.6) Splitting the Dataset

We have split the dataset into training set and test set in 80:20 ratio.

The training dataset is the one on which the model would be trained meaning it will learn the value of all the weights using iteration over the trained dataset and try to make predictions after minimising the loss.

The test dataset is the one which the model has not seen before and the closer the accuracy observed on both the train and test dataset, the better performing the model is for deployment in the practical field.

3.7.) Applying Th§2 Mod€§l

For our project, we have used the dataset to train on three different model whose literature we have surveyed so far namely:

A.) AlexNet

B.) VGG-Net

C.) ResNet

The summary of Resnet is:

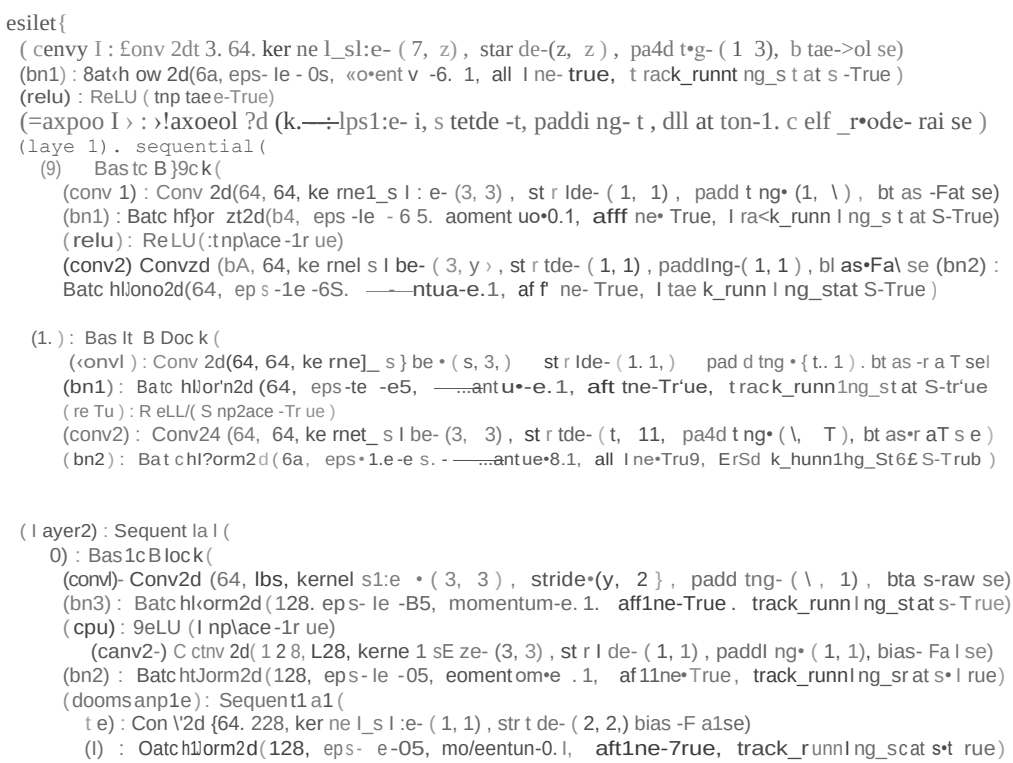


Figure 3.8.) Resnet Summary

The summary of Alexnet Model is:

```
AT ex Net (
  ( f eatures ) : Sequent ia 1(
    (0) : Conv2d ( 3, 64, kernel _s i ze- ( 11, 11) , st ride- (4, a j , padding- (2, 2) )
    (1) : ReLU(tnplace-Tru€)
    (2) : NaxPool2d(kernel_size=3, stride-2, padding=0, dilation=1, ceil_mode=False)
    (3) : Conv2d ( 64, 192, k erne1_s:t ze• (S, S), st r t de• (1, 1 ) , padding- (2, 2) )
    (4 : ReLU( inp1ace-True)
    (5) : F\axPoo 1 2d ( kerne1_s i z e-1, stride -2, paddi ng•6, d11at1on-1, cei 1_mode- False)
    (6) : Conv2d ( 192, 384, kerne1_si ze= (3, 3) , st ride = (1, 1) , paddi ng•( 1, 1)}
    (7) : ReLU(inplace=True)
    (8) : Con v2d( 3Ba, 256, kenne1 _size- ( 3, 3) , st r1de• (1, 1) , paddlng•( 1, 1) )
    (9) : ReLU( i np1ace-True )
    ( 16) : C onv 2d (256, 2 56, ke oneUs i ze- ( 3, 3 ) , st r1de -( 1, 1) , paddy ng• ( 1, 1 ) )
    ( ll ) : ReLU ( in pt a ce - T ru e )
    (12) : flaxPgol2d(kernel_size-3, stride•2, padding-0, dilation•1, ceil_mode•False)

  (avgpoo l ) : Adaptive A 'gPoo12 d(output_s 1ze•( 6, 6) )
  (c1 assi £1er ) : Sequent 1a1 (
    (B) : Dropout(p=0.5, inplace-False)
    (1) : L1near(ln_featur$S•92l6, Out_features-4e96, bias•True)
    (2) : ReLU( inp1ace=True )
    (3) : Dropout (p-0. s, 1np1ace-Fat se)
    (4 } : L1nea r ( in features =4096. out Ie at ures=4096. bias = True )
    ( S ) : ReLv( i np1ac e-True }
    (6) : L1near(in_features•4086, out_features-1000, bias•True)
  )
```

Figure 3.9.) AlexNet Summary

VGG-Net:

The summary of VGG-Net 16 is:

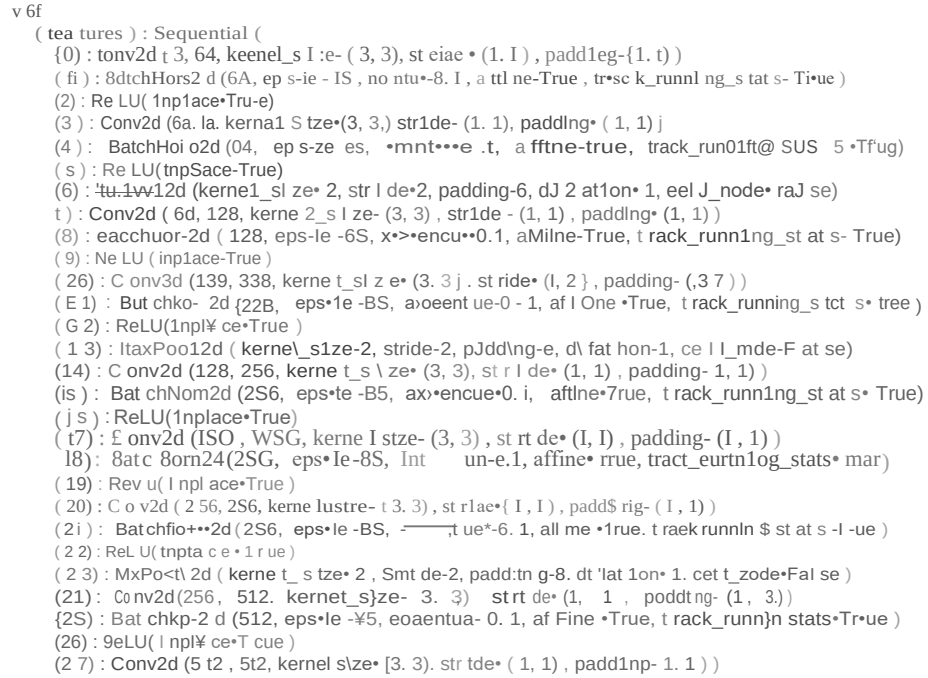


Figure 3.10.) VGG-Net architecture Summary

4. Algorithms

Once we have applied the model and applied forward propagation we would get a certain loss value on the model(deviation from the original value), it can be optimised through backpropagation, by applying different activation functions, initialisation methods, optimisation methods and batch normalisations.

4.1. Activation Functions and Initialisation Methods

Need of Activation function: Without activation functions, a neural network can only address linear relationship between the variables which we refer to as linear regression model.So to solve complex problems like image dassification, object detection, language translation we need activation function.

The representation powe f a deep Neural Network is due to its activation function.
Some of the most popular activation functions are:

- e Sigmoid/Logistic
- tanh-Hyperbolic tangent
- ReLu-Rectified Linear units
- Leaky ReLu

Logistic Activation function:
This activation function is biologically motivated representing the rate of firing in the neuron.
Mathematical representation: /(.r) = , ,' ,
This function ranges from 0 i‹› and has S-shape curve.

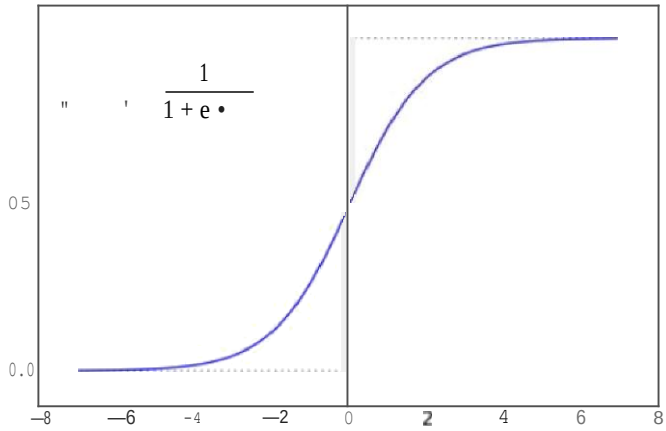


Figure 4.1.) Sigmoid Function

Relationship between logistic function and its differential:

$$f'(x) = f(x).(1 - f(x))$$

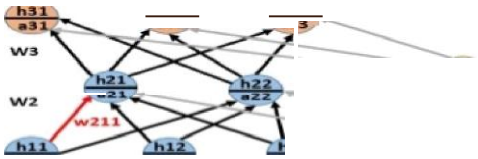
The derivative of a Logistic function helps in calculation during Backpropagation.

Reasons due to which this function has lost its popularity :

- Gradient vanish due to saturated neuron.
- Its output isn't zero centred (yields only positive value) which makes optimization harder.

Gradient Problem.

Mathemati Ily a neuron is said to be saturated when .‹) 0 *or* \
Or simply, A neuron is said to be saturated when it reaches its peak value which can be maximum Or minimum.



$$w211 = w211 - \eta \nabla w211$$

Where ∇ w211= $\frac{\partial L}{\partial}$ $\frac{\partial y}{}$

a

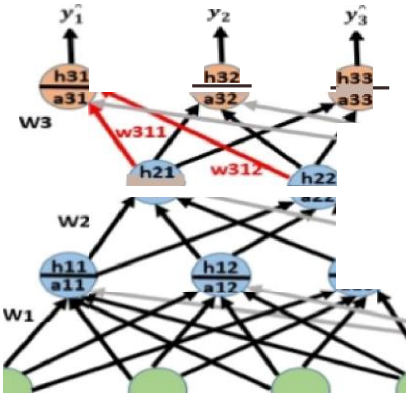
Consider the above example of a neural network in which the weight W211 needs to be updated during backpropagation using gradient descent rule. If H21 is found to be 1 then its gradient will be zero, so there is no update in the value of W211.This is known is vanishing gradient problem where the gradient of weight vanishes or goes down to zero.

Not a zero centered function:

A function is said to be zero centered if it has equal masks on both sides of the .Y — LYi.r, means it has both positive and negative values.but in case of a logistic function the output is always positive and is accumulated towards one side of the plane.

why zero centered functions?

Consider the following example of a neural network in which the weights w311 and w312 needs to be updated, and we are using logistic activation function which isn't zero centered.



B3

Common part	H 21 and h22	tJverall value

positive	Always positive	positive
negative	Always positive	negative

tanh/Hyperbolic tangent:

It is a zero centered function which ranges from — 1 /‹› 1. So tanh overcomes the non-centered problem in logistic function.

Mathematically it is represented, : /{r —,"',",", (*tanh function*)

Relationship with its derivative : /'(x) — I — (x))^2

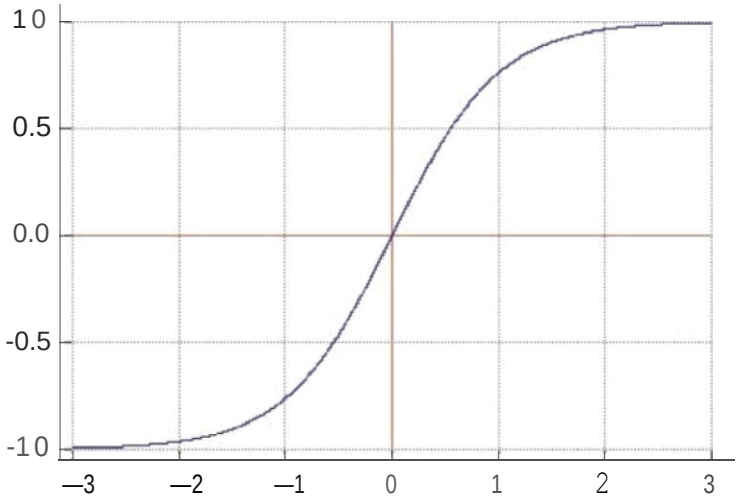


Figure 4.2.) Hyberbolic Tangent/ tanh function

- Like sigmoid, tanh also has the vanishing gradient problem.
- /‹w// iscomputationally expensive( ).

ReLU function:

This function gives output x if x (input) is positive and zero otherwise.

ReLU function is not zero centered.

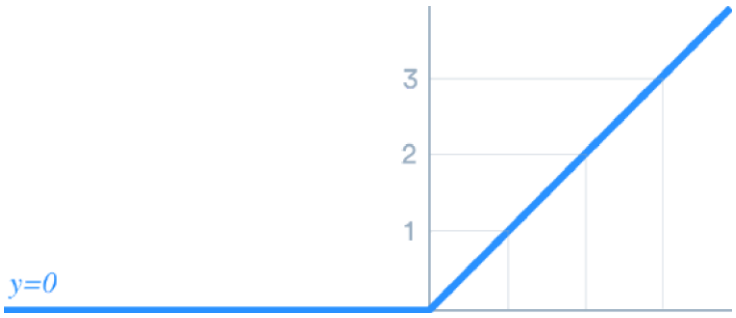


Figure 4-3-) ReLuU function

Mathematically it is represented as: *f* —— ar(0,.x)

$$f'(x) = \frac{\delta f(x)}{\delta x} = 0 \text{ if } x < 0; 1 \text{ if } x > 0$$

This function doesn't saturate in positive region.

ReLu is used in Convolution Neural Network. ReLu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. That is a good point to consider when we are designing deep neural nets.

Dying ReLU problem:

The downside for being zero for all negative values is a problem called "dying ReLU." A ReLU neuron is "dead" if it's stuck in the negative side and always outputs 0. Because the slope of ReLU in the negative range is also 0, once a neuron gets negative, it's unlikely for it to recover. Such neurons are not playing any role in discriminating the

input and is essentially useless, it might become the case where you may end up with a large part of your network doing nothing.

Other variants of ReLU

Leaky Re LU:

Mathematical representation : /(.r) - I(0.01.I,.I)

$f'(x) = 0.01 \text{ if } x < 0; 1 \text{ if } x > 0$

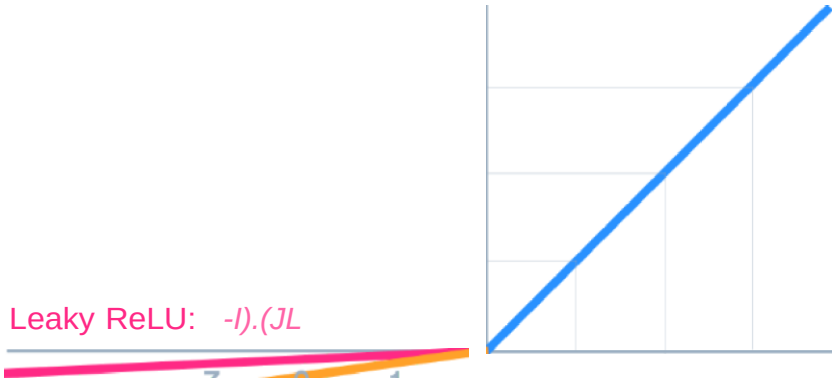


Figure 4.4.) Leaky Re LU

- This function doesn't saturate in positive or negative regions.

Initialization Methods:

the first step that should be in consideration while building a neural network is the initialization of parameters, if done correctly then optimization will be achieved in the least time otherwise converging to a minima using gradient descent will be impossible.

Types of Initialization methods:

Zero Initialization
Random Initialization
He Initialization
Xavier Initialization

Zero Initialization:
In zero initialization method, all the parameters are initialized to zero. But this problem faces Symmetry breaking problem.This problem occurs because if we initialize all the parameters with equal values, then at later instances the weights will remain equal and will get updated equally. So, zero initialization is not a good technique.

Random Initialization:
Assigning random values to weights is better than just 0 assignment. But there is one thing to keep in my mind is that what happens if weights are initialize igh values or very low values and what is a reasonable initialization of weight values.This solution is better but doesn't properly fulfil the needs so, let us see a new technique.

He initialization:

This technique is used for ReLU function.we just simply multiply random initialization:

$$w-2=\frac{random.rand(num-in,num-out)}{sqrt(num-in/2)}$$

Xavier Initialization:

This initialization technique is used for *i‹inh* ñ'J /og/.till functions.This technique is similar to He technique and random initialization is done with:

$$w-2=\frac{random.rand(num-in,num-out)}{sqrt(num-in)}$$

4.2. Batch Normalisation and Dropout

Batch normalization and drop out:

By normalizing the inputs we are able to bring all the inputs features to the same scale. In the neural network, we need to compute the pre-activation for the first neuron of the

first layer a . We know that pre-activation is nothing but the weighted sum of inputs plus bias. In other words, it is the dot product between the first row of the weight matrix and the input matrix plus bias.

Since we are computing the mean and standard deviation from a single batch as opposed to computing it from the entire data. Batch normalization is done individually at each hidden neuron in the network.

Learning Gamma v and Beta g:

g›t» *'. •.

'».• t.» +

a

The parameters Gamma and Beta are learned along with other parameters of the network. If Gamma (v) is equal to the mean (p) and Beta (g) is equal to the standard deviation(a) then the activation h final is equal to the h norm, thus preserving the representative power of the network.

Dropout:

It helps to reduce overfitting and generalization error. Dropout is a regularization technique that “drops out” or “deactivates” few neurons in the neural network randomly in order to avoid the problem of overfitting.

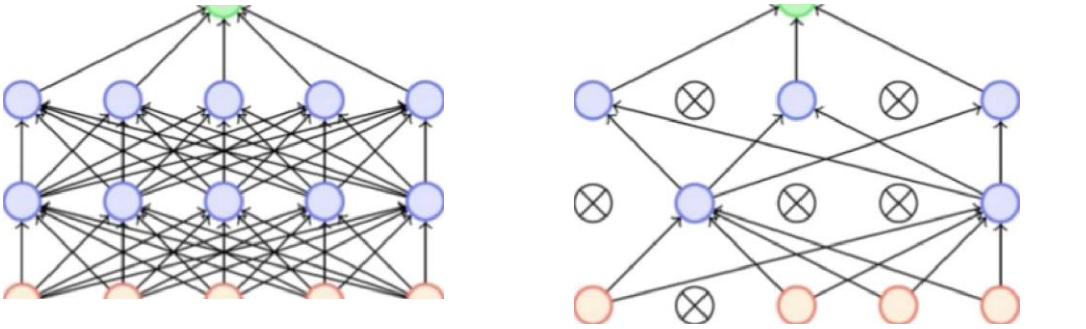


Figure 4.5.) Dropout

In the regularisation method of dropout, we try to train the model on the network with some neurons weight dropped to zero at certain layers.This decreases the computational time and complexity. We can also decide on the percentages of neurons to be dropped at which layer in the model.

4.3. Optimisation Algorithms:

We make a very regular use of optimisation algorithm in the field of machine learning to optimise various hyperparameters involved in calculating the loss function and is useful in training the model upto maximum accuracy. The hyperparameter we generally deal with are weights of each connection between layers of neurons and their biases. We generally try out various combination of optimisation algorithm, initialisation methods, activation function and loss function to optimise the loss and accuracy level.

Some of the popular optimisation algorithms are:

1. Gradient Descent

2. ADAM
3. AdaGrad
4. RMSProp
5. Adagrad

Gradient Descent avcan further be classified into Mini-batch gradient descent and Stochastic Gradient Descent.

5. Test, Result and Performance Analysis

5.1. Ratio of Dataset Splitting

We have split both the dataset of Hindi and Tamil language into 80:20 ratio. So, the model will be trained on the 80% of the dataset and its accuracy would be tested on the 20% dataset.

We can split the dataset using random split function imported from torch.utiIs.data library.

```
train_size = int(0.8 * 1en(full_data))
test size = len(full data) - train size

train_data, test_data = random_split(full_data, [train_size, test_size])
```

Figure 5.1.) Splitting the dataset

5.2. T€?Sting Loss and accuracy on R€?sNet ModE?I

We have trained the resnet model on both the dataset using the following hyperparameter.

A.) Epochs(No. of Iteration): 10

B.) Loss Function: Cross Entropy Loss Function

C.) Optimisation Algorithm: ADAM

D.) Batch Size: 64

E.) Processor: 'Cuda 0

For Hindi Dataset:

```
tensor e.ee??, de ice='cuJa:e', grab fn=<nllrossBack ar1>:
```

Figure 5.2.) Resnet Hindi Loss with Epochs

Final loss: 0.0077

Figure 5.3.) ResNet Hindi Training Accuracy

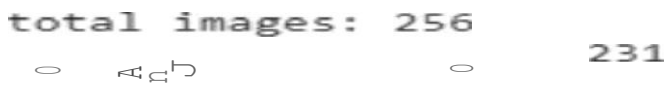


Figure 5.4.) ResNet Hindi Testing Accuracy

With Optimisation RMSprop:

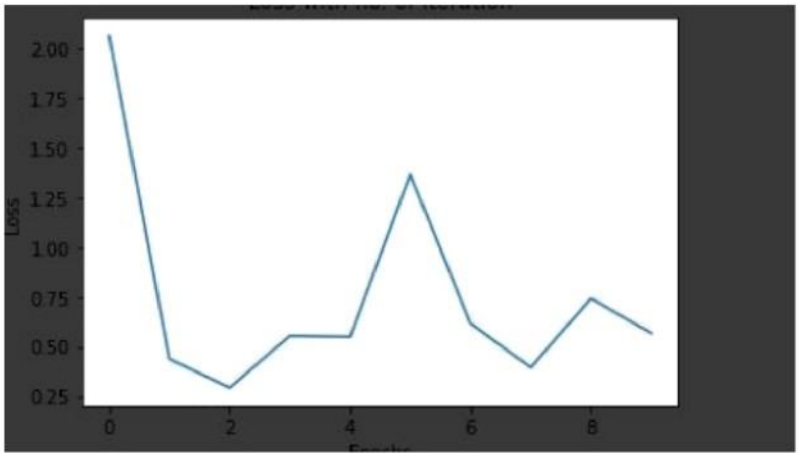


Figure 5.5.) Resnet RMSprop hindi loss

Figure 5.6.) Resnet RMSprop training accuracy

Figure 5.7.) Resnet RMSprop testing accuracy

For Tamil Language Dataset:

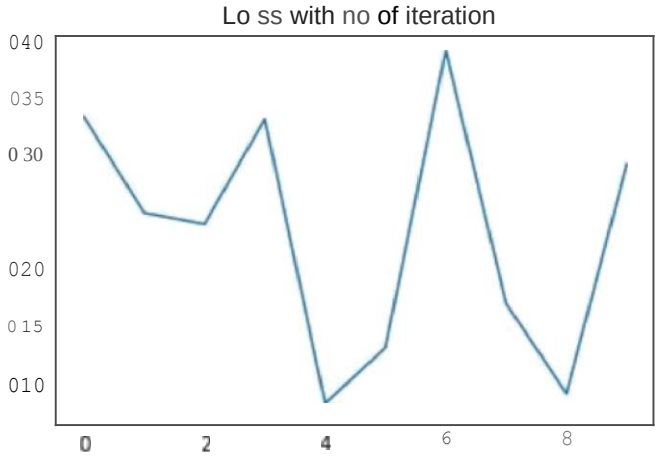


Figure 5.8.) Resnet Tamil Loss with Epochs

```
cotal images : lP2
correcr precicziors: 95S
 Acct:racy:  ?*2.5ST†87S
```

Figure 5.9.) Resnet Tamil Training Accuracy

```
-otal  images :  2oi
correct preQictio is : 234
 accuracy:   ml.4O‹52E %
```

Figure 5.10.) ResNet Tamil Testing Accuracy

5.2. Testing Loss and accuracy on AlexNRt Model

We have trained the resnet model on both the dataset using the following hyperparameter.

A.) Epochs(No. of Iteration): 12

B.) Loss Function: Cross Entropy Loss Function

C.) Optimisation Algorithm: ADAM

D.) Batch Size: 64

E.) Processor: 'Cuda 0

For Hindi Language:

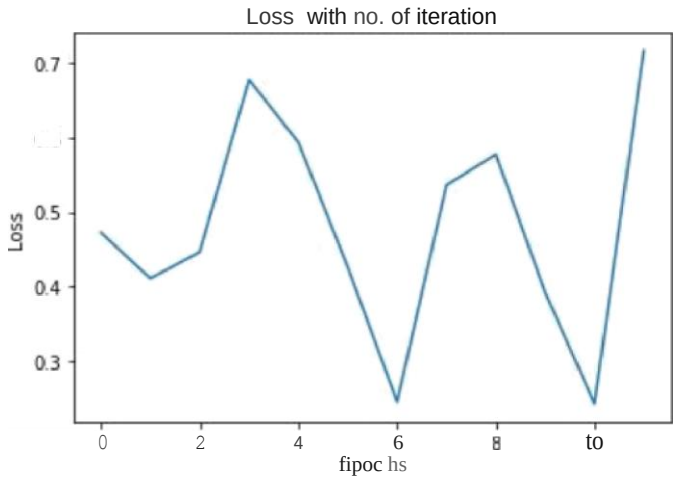


Figure 5.11.) Alexnet Hindi Loss with Epochs

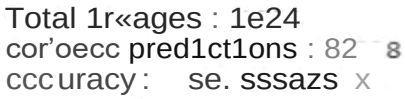


Figure 5.12.) AlexNet_Hindi_Training Accuracy

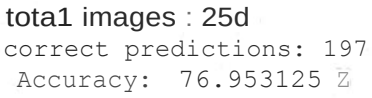


Figure 5.13.) AlexNet_Hindi_Testing Accuracy

With optimisation Algorithm RMSprop:

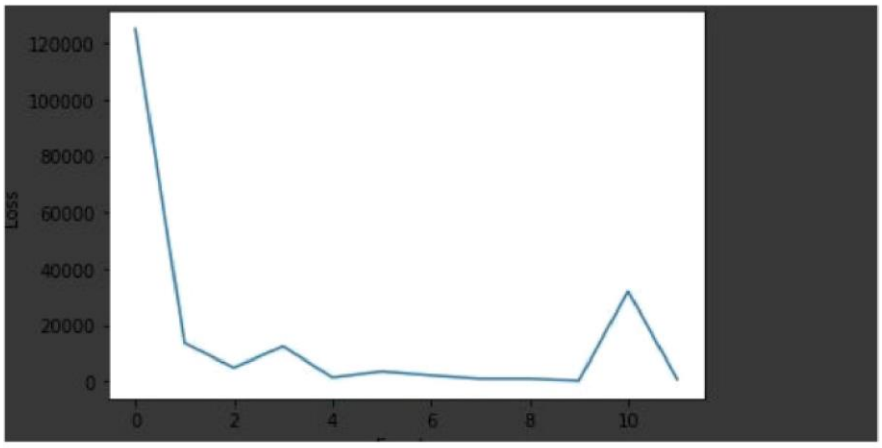


Figure 5.14.) Alexnet_RMSprop_loss

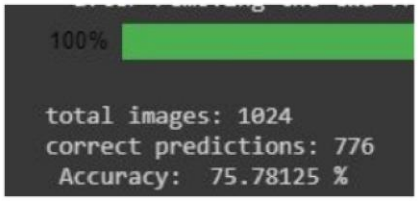


Figure 5.15.) Alexnet_RMSprop_training_error

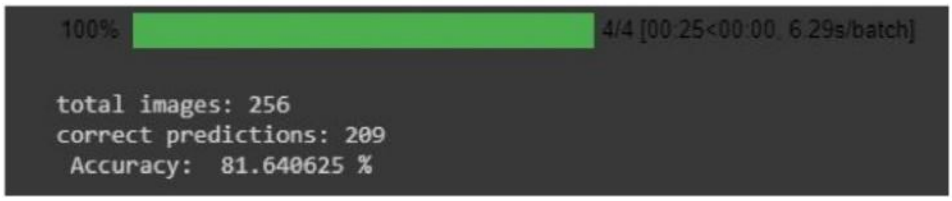


Figure 5.16.) Alexnet RMSprop test loss

For Tamil Language:

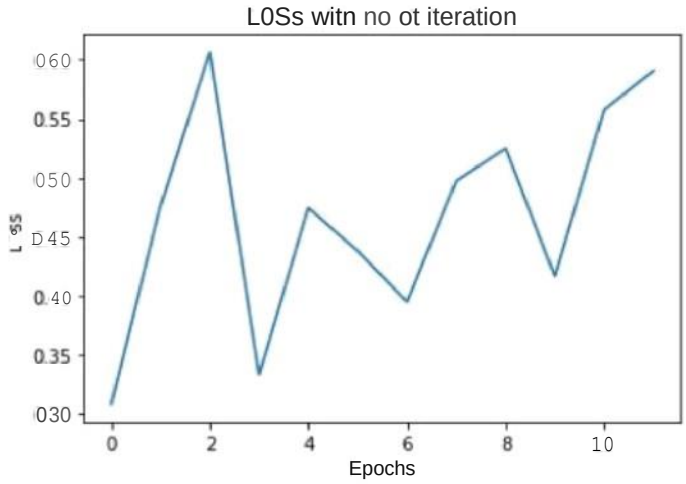


Figure 5.17.) AlexNet Tamil Loss with Epochs

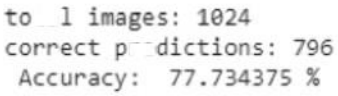


Figure 5.18.) AlexNet Tamil Training Accuracy

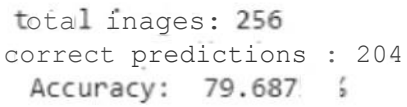


Figure 5.19.) AlexNet Tamil Testing Accuracy

5.4. TE§sting Loss and accuracy on VGG-Net ModE?I

We have trained the resnet model on both the dataset using the following hyperparameter.

A.) Epochs(No. of Iteration): 12

B.) Loss Function: Cross Entropy Loss Function

C.) Optimisation Algorithm: ADAM

D.) Batch Size: 16

E.) Processor: 'Cuda 0 '

For Hindi Language:

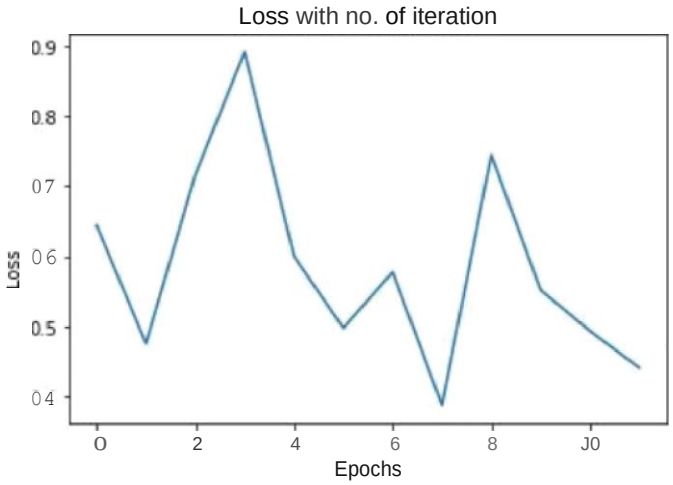


Figure 5.20.) VGGNet Hindi Loss with Epochs

```
total images: 1024
correct predictions: 796
 Accuracy:  77.734375 %
```

Figure 5.21.) VGGNet Hindi Training Accuracy

```
tota  images: 1024
correct predictions: 796
 Accuracy:  77.734375 %
```

Figure 5.22.) VGGNet Hindi Testing Accuracy

With Optimisation Algorithm RMSprop:

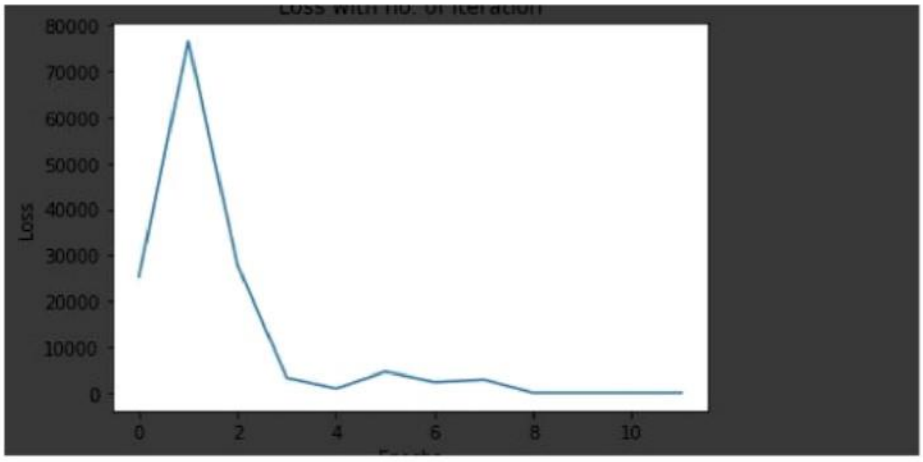


Figure 5.23.) VGGNet RMSprop loss

```
total images: 1024
correct predictions: 733
 Accuracy:  71.58203125 %
```

Figure 5.24.) VGGNet_RMSprop_train

Figure 5.25.) VGGNet RM Sprop test

For Tamil Language:

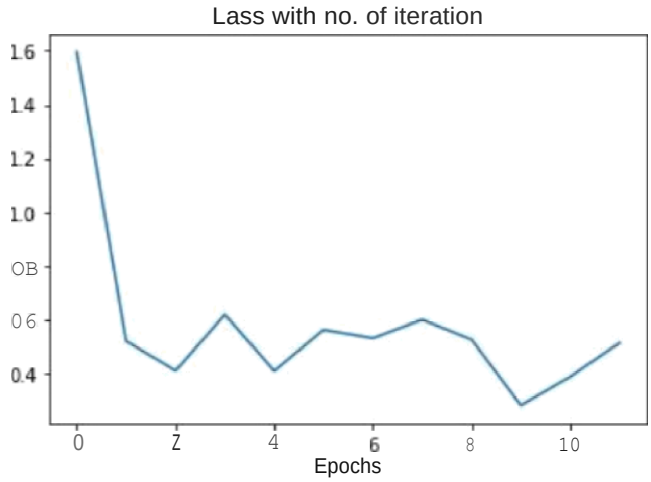


Figure 5.26.) VGGNet Tamil Loss with Epochs

```
-otel images: 1C2
correct predictions: 796
 Accuracy:  77.734375 %
```

Figure 5.27.) VGGNet Tamil Training Accuracy

```
-ota    images :  256
cor•rect predict io‹,s : 2go
 MCC nr'ac;r:   I?.t:S?5  A
```

Figure 5.28.) VGGNet Tamil Testing Accuracy

6. Conclusion and Future Work

Based on the observed loss, training accuracy and test accuracy after training three models (resnet, alexnet, vggnet) on the dataset of hindi and tamil language, we can infer from the above observation that ResNet18 models work best in case of text classification for our dataset and is more efficient than AlexNet and VGGNet.

Model	Training Accuracy	Test Accuracy
ResNet18	95.70	90.23
AlexNet	80.85	76.95
VGGNet	77.73	77.73

Table 1: Accuracy for Hindi Language

For Tamil Language:

Model	Training Accuracy	Test Accuracy
ResNet18	93.55	91.40
AlexNet	77.73	79.68
VGGNet	77.73	79.68

Table 2: Accuracy for Tamil Language

The depth of ResNet along with the concept of residual learning gives ResNet an advantage in Image Glassification over other deep learning networks.

In the next semester, we want to make to train the dataset using other models such as YOLO(You Only Look Once) and Detectron and try to improve the accuracy of the model trying different hyperparameters.

References:

1. “ Digital Recognition based on improved LeNet Convolutional Neural Network ” by Li Sun, Lishan Jia

2. “VGG-Net Architecture ” httDs://arxiv.org/pdf/1409.1556.pdf

3. “Going Deeper with Convolution ”

4. “Deep Residual Learning for Image Recognition https://www.cv-foundation.oro/openaccess/content cvpr 2016/papers/He Deep Residu al L eamino CVP R 2016 paDer.pdf

5. P orch.org

6. Forums.fast.ai

7. towardsdatascience.com

Pratyush and Varan Project Report-2

ORIGINALITY REPORT

16%	12%	4%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	towardsdatascience.com Internet Source	7%
2	www.tinymind.com Internet Source	2%
3	Submitted to Jaypee University of Information Technology Student Paper	1%
4	medium.com Internet Source	1%
5	Submitted to Higher Education Commission Pakistan Student Paper	<1%
6	qims.amegroups.com Internet Source	<1%
7	Submitted to University College London Student Paper	<1%
8	Submitted to Asia Pacific Instutute of Information Technology Student Paper	<1%

9	Logan R. Vallandingham, Quan Yu, Nakul Sharma, Jo W. Strandhagen, Jan Ola Strandhagen. "Grocery retail supply chain planning and control: Impact of consumer trends and enabling technologies", IFAC-PapersOnLine, 2018 Publication	<1%
10	aidreams.co.uk Internet Source	<1%
11	Submitted to UT, Dallas Student Paper	<1%
12	Submitted to Jawaharlal Nehru University (JNU) Student Paper	<1%
13	Submitted to CSU, San Jose State University Student Paper	<1%
14	everything.explained.today Internet Source	<1%
15	kr.mathworks.com Internet Source	<1%
16	www.mdpi.com Internet Source	<1%
17	Submitted to Instituto de Empress S.L. Student Paper	<1%
18	Submitted to De Montfort University Student Paper	<1%

19	www.inderscienceonline.com Internet Source	<1%
20	Submitted to University of Newcastle upon Tyne Student Paper	<1%
21	"Communications, Signal Processing, and Systems", Springer Science and Business Media LLC, 2020 Publication	<1%
22	Submitted to University of Sheffield Student Paper	<1%
23	www.tutorialspoint.com Internet Source	<1%
24	Submitted to National Technical University of Athens Student Paper	<1%
25	Submitted to Bournemouth University Student Paper	<1%
26	Submitted to University of Bristol Student Paper	<1%
27	Babak Ehteshami Bejnordi, Mitko Veta, Paul Johannes van Diest, Bram van Ginneken et al. "Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer", JAMA, 2017	<1%

Publication

28	Hisham El-Amir, Mahmoud Hamdy. "Deep Learning Pipeline", Springer Science and Business Media LLC, 2020 Publication	<1%
29	Submitted to Informatics Education Limited Student Paper	<1%
30	Submitted to Turun yliopisto Student Paper	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		