Sardar Patel Institute of Technology,Mumbai

Department of Electronics and Telecommunication Engineering

T.E. Sem-V (2018-2019)

ETL54-Statistical Computational Laboratory

Lab-7:Image Classification using Artificial Neural Network (ANN)

Name:Manish Dsilva **Roll No.15**

Objective: To classify the images using ANN

Outcomes:

1. To build the neural network (Input/Hidden/Output Layers)

2. To install keras and tensorflow library

3. To define the NN model to classify the images

4. To visualize the image dataset

5. Evaluate the ANN model with performance metrics

System Requirements:

1.Ubuntu OS with Anaconda Python framework with Keras, Tensorflow library.
2. MNIST Fashion-MNIST database of fashion articles, MNIST database of handwritten digits

Introduction to ANN: An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output.

ANNs are considered nonlinear statistical data modeling tools where the complex relationships between inputs and outputs are modeled or patterns are found.

ANN is also known as Artificial neural network.

ANNs have three layers that are interconnected. The first layer consists of input neurons. Those neurons send data on to the second layer, which in turn sends the output neurons to the third layer.

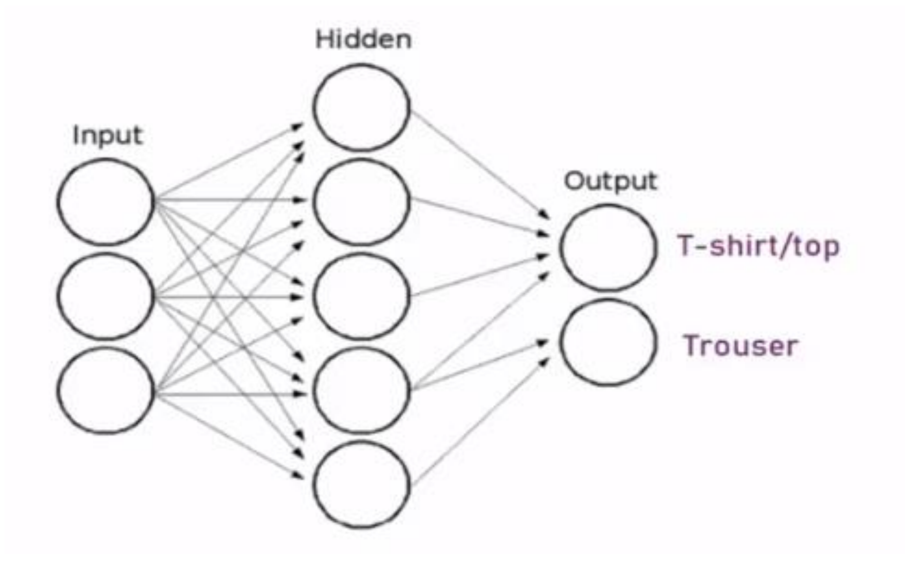


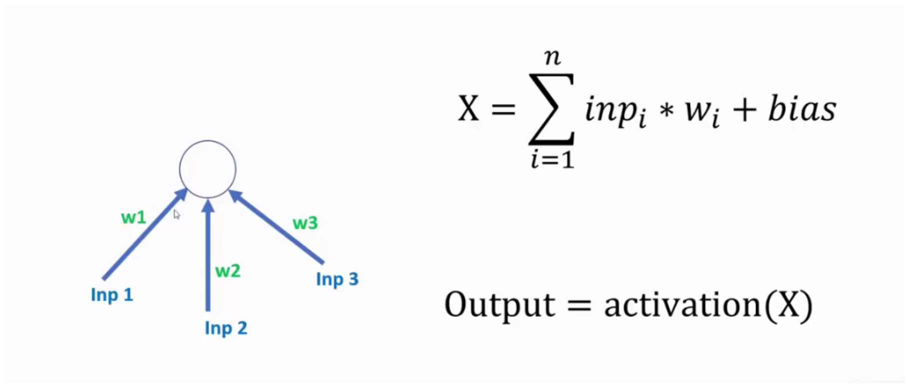
Figure-1: ANN

Elements of a Neural Network :
Input Layer :- This layer accepts input features. It provides information from the outside world to the network, no computation is performed at this layer, nodes here just pass on the information(features) to the hidden layer.
Hidden Layer :- Nodes of this layer are not exposed to the outer world, they are the part of the abstraction provided by any neural network. Hidden layer performs all sort of computation on the features entered through the input layer and transfer the result to the output layer.
Output Layer :- This layer bring up the information learned by the network to the outer world.

Activation function: Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to **introduce non-linearity** into the output of a neuron.
We know, neural network has neurons that work in correspondence of *weight, bias* and their respective activation function. In a neural network, we would update the weights and biases of the neurons on the basis of the error at the output. This process is known as *back-propagation*. Activation functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases.

Need of Non-linear activation functions :

A neural network without an activation function is essentially just a linear regression model. The activation function does the non-linear transformation to the input making it capable to learn and perform more complex tasks.



w1
w2
w3
Inp 1
Inp 2
Inp 3
$X = \sum_{i=1}^{n} inp_i * w_i + bias$
$\text{Output} = \text{activation}(X)$

Figure-2: Activation Function

Activation Functions used in ANN: Step, Sigmoid and Relu

About Keras: Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.

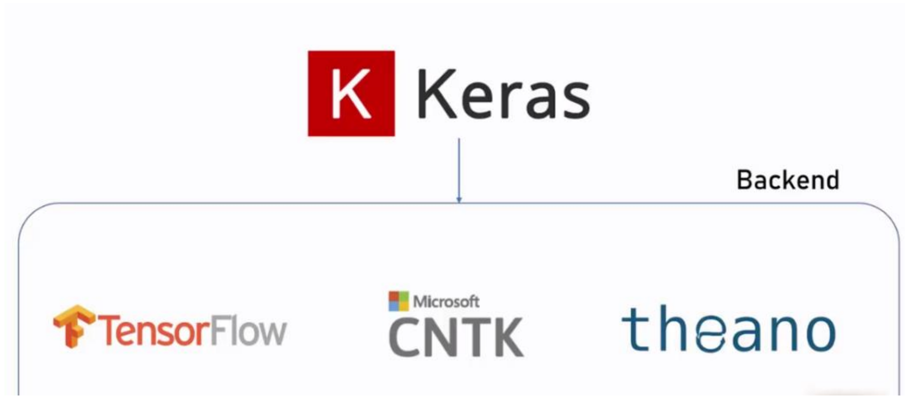


Figure-3: Keras with Backend

Procedure:

Step-1: Install the keras and tensorflow

```
$pip install --upgrade pip

$pip install keras

$pip install tensorflow

$conda list|grep keras

$conda list|grep tensorflow
```

Step-2: Check keras library with tensorflow as backend

```
$python

>>>import keras

>>>keras.backend.backend()
```

Step-3: Open jupyter-notebook

Go to new and select Python3 version

```
#Import library

import pandas as pd

import numpy as np

import scipy as sp

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline
```

Task-1: Load the **Fashion-MNIST database of fashion articles**

Refer:[1]https://keras.io/datasets/

Dataset of 60,000 28x28 grayscale images of 10 fashion categories, along with a test set of 10,000 images. This dataset can be used as a drop-in replacement for MNIST. The class labels are:

Label	**Description**
0	T-shirt/top
1	Trouser

Label	**Description**
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

#Dataset partitions-Train and Test images dataset

#Usage:

from keras.datasets import fashion_mnist

(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()

Returns:

2 tuples:

x_train, x_test: uint8 array of grayscale image data with shape (num_samples, 28, 28).

y_train, y_test: uint8 array of labels (integers in range 0-9) with shape (num_samples,).

#View the image sizes

x_train.shape

x_test.shape

#Use Matplotlib to visualize the images

plt.matshow(x_train[0])

plt.matshow(x_train[1])

#Check with image dataset labels

```
y_train[0]

y_train[1]
```

# Normalize the image dataset (0-1)

```
x_train=x_train/255

x_test=x_test/255
```

#Build the neural network

```
from keras.models import Sequential

from keras.layers import Dense, Activation, Flatten

model=Sequential()

model.add(Flatten(imput_shape=[28,28]

model.add(Dense(20,activation='relu')

model.add(Dense(10,activation='softmax')
```

# Compile the model and summary

```
model.compile(loss="sparse_categorical_crossentropy",optimizer="adam",
metrics=["accuracy"])

model.summary()
```

#Train the model

```
model.fit(x_train,y_train,epoch=5)
```

#Prediction

```
x_test.shape

#See the image

plt.matshow(x_test[0])

#Check the label

y_test[0]

ypred=model.predict(x_test)
```

#Check the fist prediction

ypred[0] #Dense(10) output 10

#Verify with numpy

np.argmax(yped[0])

#Evaluate the model

model.evaluate(x_test,y_test)

#Tune the model (Hyperparameter tuning)

#Rebuild the model and chek the performance

Result Analysis:

Task-1: Load the MNIST Fashion database

1. Training time = 43s
2. Prediction time = 0.469s
3. [Loss, Accuracy] = [0.4182774677276611, 0.8536]

Task-2: Load the MNIST database of handwritten digits

1. Training time = 51s
2. Prediction time = 0.412s
3. [Loss, Accuracy] = [0.1481,0.9579]

What is Deep Learning and why is it so popular?

Practically, ***Deep Learning*** *is a subset of* ***Machine Learning*** that achieves great power and flexibility by learning to represent the world as nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones.Elaborately, a deep learning technique learn categories incrementally through it's hidden layer architecture, defining low-level categories like letters first then little higher level categories like words and then higher level categories like sentences. In the example of image recognition it means identifying light/dark areas before categorizing lines and then shapes to allow face recognition. Each neuron or node in the network represents one aspect of the whole and together they provide a full representation of the image. Each node or hidden layer is given a weight that represents the strength of its relationship with the output and as the model develops the weights are adjusted.

Is Deep Learning just a hype or does it have real-life applications?

The hype is real. Artificial Intelligence (AI) and Machine Learning (ML) is all over the media, and everyone wants to be involved in the technology race. Progress during the last few years has been absolutely tremendous, and you have probably heard claims such as “AI is the new electricity” and “AI will completely revolutionize our society”. I will not comment on those statements, but what we can all safely agree on, is that there is definitively a lot of interest concerning these technologies. However, all this attention begs the important question: can it really live up to the hype?

Within certain areas, the technology has caught up to, and even surpassed the hype. For example, within image recognition, the task of identifying objects and extracting information from images, AI is now going beyond human level performance (i.e., machines are actually getting better than humans in identifying objects and images).

What is the difference between Deep Learning and Machine Learning?

The key difference between deep learning vs machine learning stems from the way data is presented to the system. Machine learning algorithms almost always require structured data, whereas deep learning networks rely on layers of the ANN (artificial neural networks).Machine learning algorithms are built to “learn” to do things by understanding labeled data, then use it to produce further outputs with more sets of data. However, they need to be retrained through human intervention when the actual output isn’t the desired one.Deep learning networks do not require human intervention as the nested layers in the neural networks put data through hierarchies of different concepts, which eventually learn through their own errors. However, even these are subject to flawed outputs if the quality of data isn’t good enough.Data is the governor here. It is the quality of data which ultimately determines the quality of the result.

What are the prerequisites for starting out in Deep Learning?

Most of machine learning and AI courses need good math background. You should have good knowledge of calculus,linear algebra, stats and probability. But this deep learning course just needs a little bit knowledge of linear algebra and calculus. Probability is not much used. Same is with stats. Other than that you should know how to use Python

Is it necessary to have a research background in Deep Learning to start out in it?

No, a PhD is not a mandatory requirement to make a career in deep learning. You can learn, experiment, and build up your work experience portfolio without going to university. The emphasis for any job or role is usually on demonstrating your competence, and not on your degree, per se.Having said that, a PhD in a specific field (like linguistics for NLP) will definitely accelerate your path if you choose to combine that with deep learning.

Which Tools/Languages should I prefer to build Deep learning models?

Recommended to use Python, because of its robust ecosystem for machine learning. The python ecosystem comprises of developers and coders who are providing open source libraries and support for the community of python users. This makes the task of writing complex codes for various algorithms much easier and the techniques easier to implement and experiment with.Also, Python being a more generalized programming language, can be used for both the development and implementation. This greatly simplifies the transition from development to operations. That is, a deep learning product that can predict the price of flight tickets, can not only be developed in python but can also be attached with your website in the same form. This is what makes Python a universal language.Besides this, I would suggest that beginner's use high level libraries like Keras. This makes experimentation easier by providing abstraction to the unnecessary information that is hidden under the algorithms. And giving access to the parameters that can be tweaked to enhance the performance of such models. Let us understand this with an example:When you press the buttons on a television remote, do you need to care about the background processes that are happening inside the remote? Do you need to know about what signal is being sent out for that key, or how is it being amplified?

No, right?

Because maybe an understanding of these processes is required for a physicist but for a lame man sitting in his bedroom, it is just an information overload.There are also other contenders apart from Python in the deep learning space such as R, Julia, C++, and Java. For alternatives of libraries, you can check out TensorFlow, Pytorch, Caffe2, DL4J, etc. We should stay updated with their developments as well.If you are not well versed with programming, there are also a few GUI based softwares, that require no coding, to build deep learning models, such as Lobe or Google's AutoML, among others.

Why are GPUs necessary for building Deep Learning models?

When you train a deep learning model, two main operations are performed:

- Forward Pass
- Backward Pass

In forward pass, input is passed through the neural network and after processing the input, an output is generated. Whereas in backward pass, we update the weights of neural network on the basis of error we get in forward pass.Both of these operations are essentially matrix multiplications. A simple matrix multiplication can be represented by the image belowHere, we can see that each element in one row of first array is multiplied with one column of second array. So in a neural network, we can consider first array as input to the neural network, and the second array can be considered as weights of the network.This seems to be a simple task. Now just to give you a sense of what kind of scale deep learning – VGG16 (a convolutional neural network of 16 hidden layers which is frequently used in deep learning applications) has ~140 million parameters; aka weights and biases. Now think of all the matrix multiplications you would have to do to pass just one input to this network! It would take years to train this kind of systems if we take traditional approaches.We saw that the computationally intensive part of neural network is made up of multiple matrix multiplications. So how can we make it faster?We can simply do this by performing all the operations at the same time instead of doing it one after the other. This, in a nutshell, is why we use GPU (graphics processing units) instead of a CPU (central processing unit) for training a neural network.

When (and where) to apply Neural Networks ?

Deep Learning have been in the spotlight for quite some time now. Its “deeper” versions are making tremendous breakthroughs in many fields such as image recognition, speech and natural language processing etc.Now that we know it is so impactful; the main question that arises is when to and when not to apply neural networks? This field is like a gold mine right now, with many discoveries uncovered everyday. And to be a part of this “gold rush”, you have to keep a few things in mind:

- **Firstly, deep learning models require clear and informative data (and mostly big data) to train.** Try to imagine deep learning model as a child. It first observes how its parent walks. Then it tries to walk on its own, and with its every step, the child learns how to perform a particular task. It may fall a few times, but after few unsuccessful attempts, it learns how to walk. If you don’t let it, it might not ever learn how to walk. The more exposure you can provide to the child, the better it is.
- **It is prudent to use Deep Learning for complex problems such as image processing.** Deep Learning algorithms belong to a class of algorithms called representation learning algorithms. These algorithms break down complex problems into simpler form so that they become understandable (or “representable”). Think of it as chewing food before you gulp. This would be harder for traditional (non-representation learning) algorithms.
- **When you have an appropriate type of deep learning to solve the problem.** Each problem has its own twists. So the data decides the way you solve the problem. For example, if the problem is of sequence generation,

recurrent neural networks are more suitable. Whereas, if it is image related problem, you would probably be better of taking convolutional neural networks for a change.

- **Last but not the least, hardware requirements are essential for running a deep neural network model.** Neural nets were “discovered” long ago, but they are shining in the recent years for the main reason that computational resources are better and more powerful. If you want to solve a real life problem with these networks, get ready to buy some high-end hardware!

Conclusion:

- Artificial Neural Networks are used for Machine Learning/Deep Learning
- They are used to build both Classification and Regression problems
- They are Computationally heavy and hence require a lot of train time
- Neural networks are used for Facial/Voice Recognition, Medical Scans
- Keras has provided us a user-friendly environment for using TensorFlow

References:

[1] Keras Datasets

https://keras.io/datasets/

[2]Keras Tutorial: How to get started with Keras, Deep Learning, and Python

https://www.pyimagesearch.com/2018/09/10/keras-tutorial-how-to-get-started-with-keras-deep-learning-and-python/

[3]https://www.analyticsvidhya.com/blog/2018/05/deep-learning-faq/