CS5720: Neural Network & Deep Learning

Project Proposal + Increment

1.Project Proposal

Project Title:

Handwritten Text to Digital Text Conversion using various Deep Learning Models

Team Members:

SHINY SHERLY KATURU – 700744314

BHAVYASRI MADDINENI – 700748499

SNEHA LATHA KUSUMA – 700745374

2. Goals and Objectives:

Motivation:

As the digitalization is everywhere Hand-written text is converted into digital text. This conversion provides solutions to the various problems such as physical damage of the document, accessing the document from anywhere around the world, transferring the document and security related problems. In this conversion Handwritten Text Recognition (HTR) or Handwriting Recognition (HWR) plays an important role to recognize the handwritten text from the input scanned images and converts it into digital format. The input is given from the sources such as notebooks, documents, forms, photographs and other devices.

Significance:

The capacity to recognize and understand legible handwritten input from sources including paper

documents, touch screens, photo graphs, etc. is known as handwriting detection. Recognition of handwritten text is one type of pattern recognition. The classification of data based on previously acquired knowledge or on statistical data extrapolated from patterns and/or their representation is known as pattern recognition. This technique will be used to identify writings in various formats. The evolution of handwriting has led to the appearance of many different types of handwritten characters, including digits, numerals, cursive writing, symbols, and scripts in both English and other languages. Several applications where it is important to handle huge volumes ofhandwritten data, such as the interpretation of amounts on bank checks, document analysis, and signature verification, can benefit greatly from the automatic recognition of handwritten text. In order to make processing documents easier, a computer that can read documents or data is required.

Objectives:

The major goal is to identify handwritten text in online documents, including letters, words, lines, and paragraphs. The field of handwriting recognition has seen a lot of effort, and there have been several reviews.

- Character Extraction: It involves scanning of image and then individual character contained is to be extracted.
- Character Recognition: Once the individual characters are extracted, the computer outputs the corresponding digital character.
- Feature Extraction: In this step the programmer must determine the properties that may appear to be important.

Features:

• While the traditional techniques involve the segmentation or characters, the modern techniques involve the segmentation of lines.

• Focuses on machine learning techniques to learn the features.

• Use of Convolution Neural Networks to extract features

• Use of Recurrent Neural Networks to have overlapping windows of the text image several times.

Increment Guidelines:

1. Datasets:

To load the dataset, we use:

SCIKIT LEARN'S DIGIT DATASET

```
sklearn.datasets.load_digits(*, n_class=10, return_X_y=False, as_frame=False)
```

The dataset contains 8x8 image of a digit.

Table 1 sklearn digits dataset

Classes	10
Samples per Class	~ 180
Total Samples	1797
Dimensionality	

	64
Features	Integers 0 – 16

KERAS'S MNIST DATASET

To load the dataset, we use:

```
tf.keras.datasets.mnist.load_data(path="mnist.npz")
```

The dataset contains 28x28 image of a digit.

Table 4.3.2.1 keras mnist dataset

Classes	10
x_train	(60000, 28, 28)
y_train	(60000,)
x_test	(10000, 28, 28)
y_test	(10000,)

EXTRA KERAS'S EMNIST BALANCED DATASET

This dataset contains digits-, upper- and lower-case handwritten letters.

```
from extra-keras-datasets import emnist
(input_train, target_train), (input_test, target_test) = emnist.load_data(type='balanced')
```

Table 4.3.3.1 extra keras mnist balanced type dataset

Train	112800
Test	18800
Total	131600
Classes	47

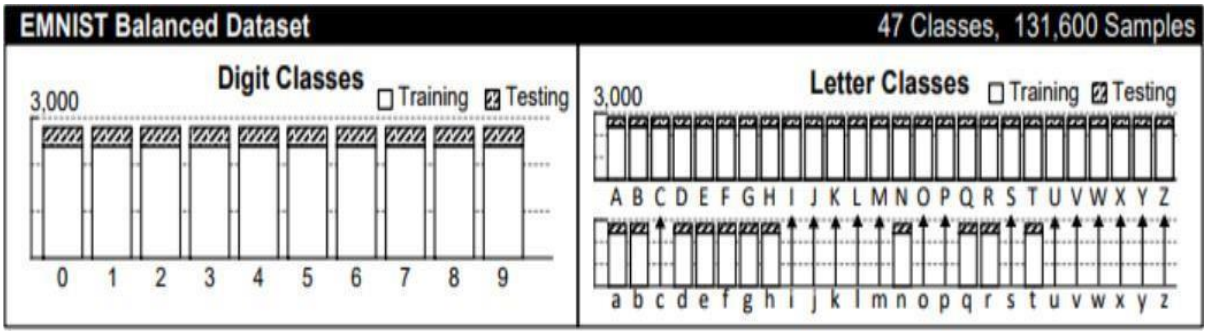


Figure 4.3.3.1 extra keras emnist balanced type dataset

Detailed Design of Features:

OCR technology achieves the accuracy greater than 99% for the typed characters in the input images with high quality. But it provides less accuracy when there are different types of handwritings, differences in spaces and irregularities of handwriting. Thus, the accuracy that is provided by the OCR systems for

handwritten characters is less when compared to the accuracy that is provided by the OCR systems for typed characters. OCR system does not contain many tools to handle handwriting recognition. As the handwritings differ from person to person, traditional OCR system cannot recognize everyone's handwriting. To recognize all the different handwritings successfully complex deep learning algorithms are to be used.

The recognition of handwritten text highly depends on Neural Networks. By using these algorithms the performance of handwriting recognition tools can be increased.

Analysis:

Implementation:

Firstly, We have used Scikit Learn Digit Dataset to perform different models like Multinomial Logistic Regression, Support Vector Machine, Decision Tree.

Secondly, We have performed Convolution Neural Networks on Kera's Mnist Dataset, Decision Tree Model is performed using Kera's Mnist Dataset.

And we have performed Multinomial Logistic Regression on Extra Kera's Mninst Dataset.

Based on the accuracy we will select the best model with high accuracy on the large dataset.

Preliminary Results:

- Multi Class Logistic Regression: Accuracy is 97.22%
- Convolution Neural Network: Accuracy is 98.31%
- Decision Tree: Accuracy is 100%

Project Management

Implementation Status Report:

Work Completed:

Description:

Multi Class Logistic Regression:

Multiclass Logistic Regression / Multinomial Logistic Regression generalizes the classification of Logistic Regression to Multiclass, i.e., with more

than binary outcome.

The ML model of this type predicts the probabilities for different outcomes. Other names of this model:

- polytomous LR
- multiclass LR
- softmax regression
- multinomial logistic
- maximum entropy
- conditional maximum entropy

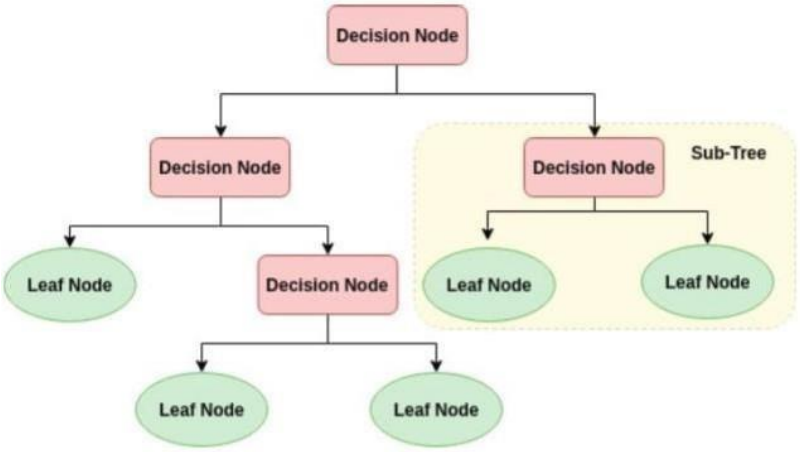
Convolution Neural Network:

It is a special type of feed-forward artificial neural network in which the connectivity pattern between the neurons is inspired by the visual cortex. The data processing in this network occurs through multiple layers of arrays. Image recognition and face recognition uses this type of neural network. CNN takes the two-dimensional array as input and directly operates on the images without focusing on the feature extraction.

Decision Tree:

Decision Tree is used to solve the problems on regression and classification. It can handle high dimensional data with good accuracy. It is very easy to understand popular classification algorithms and interpret. A decision tree is a flowchart like tree structure, each node represents the features and each edge represents the decision. The classification is a two-step process, learning step and prediction step. In the learning step, the model is developed based on given

training data and prediction step, the model is used to predict the response for given data.



Responsibilities:

Sneha Latha Kusuma:

Multi Class Logistic Regression:

- Importing the digits dataset from sklearn

```
from sklearn.datasets import load_digits
```

- Loading the digits dataset to a data frame

```
1 d = load_digits()
2 dir(d)
```

```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']
```

- Importing train test split method from sklearn
- Splitting the total dataset into train and test dataset

```
1 from sklearn.model_selection import train_test_split
```

```
1 x_train, x_test, y_train, y_test = train_test_split(d.data, d.target, test_size=0.2)
```

- Importing Linear model for Logistic Regression from sklearn
- Creating a model for Logistic Regression
- Training the model

```
1 from sklearn import linear_model
```

```
1 model = linear_model.LogisticRegression()
```

```
1 model.fit(x_train, y_train)
```

- Checking the score of model
- Predicting some of the test data
- Comparing the predicted output to actual output

```
LogisticRegression()
```

```
1 model.score(x_test, y_test)
```

```
0.9722222222222222
```

```
1 model.predict(x_test[:5])
```

```
array([9, 6, 0, 6, 7])
```

```
1 y_test[0:5]
```

```
array([9, 6, 0, 6, 7])
```

Bhavyasri Maddineni:

Convolution Neural Network:

Three basic ideas used by this network are local respective fields, convolution, pooling.

- Importing necessary libraries
- Loading mnist dataset
- Data Preprocessing

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
```

```
mnist=tf.keras.datasets.mnist
(x_train, y_train),(x_test, y_test)=mnist.load_data()
x_train_new=x_train.reshape(-1,28,28,1)
x_test_new=x_test.reshape(-1,28,28,1)
```

```
x_train_new=x_train_new/255.0
x_test_new=x_test_new/255.0
```

- Creating Convolution Neural Network Model
- Adding Layers
- Compiling the model
- Training the model

```python
from tensorflow import keras
model=keras.Sequential()
model.add(keras.layers.Conv2D(filters=32,kernel_size=(3,3),activation="relu",input_shape=(28,28,1)))
model.add(keras.layers.MaxPooling2D((2,2)))
model.add(keras.layers.Dropout(0.2))
model.add(keras.layers.Conv2D(filters=64,kernel_size=(3,3),activation="relu"))
model.add(keras.layers.MaxPooling2D((2,2)))
model.add(keras.layers.Dropout(0.5))
model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(250,activation="sigmoid"))
model.add(keras.layers.Dense(10,activation="softmax"))
model.compile(
optimizer="adam",
loss="sparse_categorical_crossentropy",
metrics=["accuracy"]
)
model.fit(x_train_new,y_train,epochs=3)


```

```
Epoch 1/3
1875/1875 [==============================] - 36s 19ms/step - loss: 0.1955 - accuracy: 0.9419
Epoch 2/3
1875/1875 [==============================] - 45s 24ms/step - loss: 0.0732 - accuracy: 0.9766
Epoch 3/3
1875/1875 [==============================] - 49s 26ms/step - loss: 0.0542 - accuracy: 0.9831

<tensorflow.python.keras.callbacks.History at 0x232e9c9bac0>
```

```
1 model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 13, 13, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
dropout_1 (Dropout)          (None, 5, 5, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 1600)              0
_________________________________________________________________
dense (Dense)                (None, 250)               400250
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2510
=================================================================
Total params: 421,576
Trainable params: 421,576
Non-trainable params: 0
```

- Predicting few elements of test dataset
- Comparing predicted output to the actual output

```
1 import numpy as np
2 y_predicted = [np.argmax(arr) for arr in model.predict(x_test_new)]
3 y_predicted[:5]
```

```
[7, 2, 1, 0, 4]
```

```
1 y_test[:5]
```

```
array([7, 2, 1, 0, 4], dtype=uint8)
```

Shiny Sherly Katuru:

Decision Tree:

- Importing required libraries
- Importing the digits dataset from sklearn

- Loading the digits dataset to a data frame

```
import pandas as pds
from sklearn.datasets import load_digits
```

```
df=load_digits()
dir(df)
```

```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']
```

- Data Preprocessing
- Importing train test split method from sklearn
- Splitting the total dataset into train and test dataset

```
x=df.data
y=df.target
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

- Importing Tree for Decision Tree Classifier from sklearn
- Creating a model for Decision Tree Classifier
- Training the model
- Checking the score of model

```
from sklearn import tree
```

```
model = tree.DecisionTreeClassifier()
model.fit(x_train,y_train)
```

```
DecisionTreeClassifier()
```

```
model.score(x_train,y_train)
```

```
1.0
```

- Predicting some of the test data
- Comparing the predicted output to actual output

```
1 model.predict(x_test[:5])
```

```
array([7, 1, 5, 0, 8])
```

```
1 y_test[:5]
```

```
array([7, 1, 5, 0, 8])
```

Contributions:

1. Sneha Latha Kusuma has contributed 20% of the work completed.
2. Bhavyasri Maddineni has contributed 30% of the work completed.
3. Shiny Sherly Katuru has contributed 30% of the work completed.

Work to be Completed:

The model is working with a good accuracy. We would like to add more epochs to get even more better accuracy for the model. We have done with multinomial logistic regression, convolution neural networks, decision tree. Furthermore, by comparing the accuracies, we decided to add two more algorithms such as random forest and support vector machine for the model.

Description:

Support Vector Machine:

Support Vector Machine is a popular supervised algorithm which can be used for classification and regression problems. It is primarily used for classification problems in machine learning. SVM algorithm creates the best line or a decision boundary to segregate n-dimensional space into classes. So, in the future we can easily include the new data points in the correct category. This is a hyperplane which is known as a best decision boundary.

It chooses the extreme points or vectors which helps in creating the hyperplane. The extreme vectors are known as support vectors. Hence this algorithm is known as Support Vector Machine. Face detection, image classification and text categorization use this algorithm.

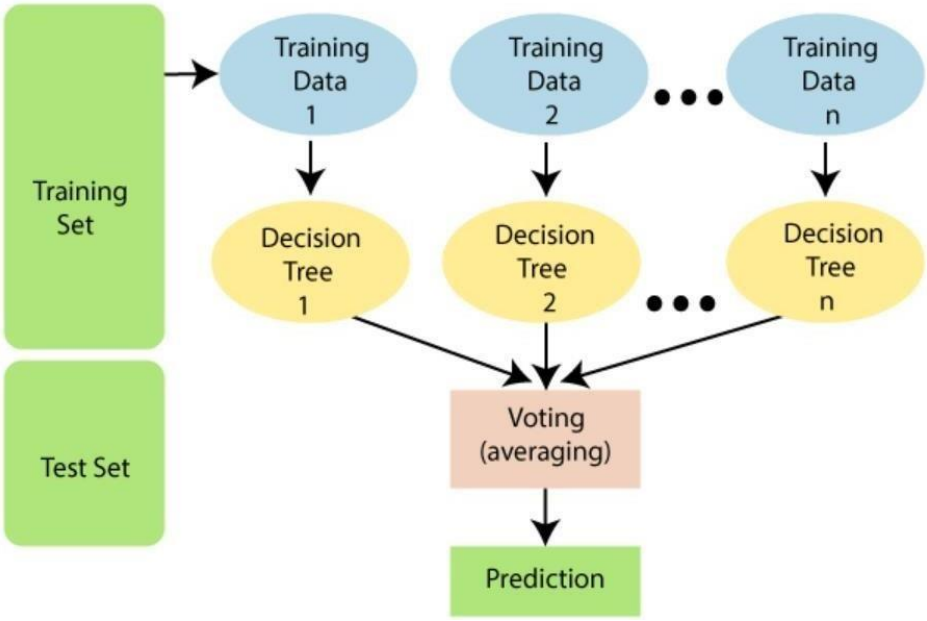
Types of SVM:

- Linear SVM: In this we use linearly separable data. If a single straight line can separate the dataset into two classes, it is known as linearly separable data. Classifier is known as linear SVM classifier.
- Non-linear SVM: In this we use non-linearly separable data. If a single straight line cannot separate the dataset into two classes, it is known as non-linearly separable data. Classifier is known as Non-linear SVM classier.

Random Forest:

Random Forest Algorithm is a machine learning algorithm which belongs to the supervised learning technique. In Machine learning Classification and Regression problems uses this algorithm. This is based on a process which combines multiple classifiers to solve a complex problem and to improve the model performance. Random Forest is a classifier with a number of decision trees on different subsets of the dataset and the average is taken to improve the predictive accuracy of the dataset.

It predicts the output based on the majority votes of predictions of each tree without depending on one decision tree. As the number of trees increases in the forest the accuracy also increases and prevents the overfitting problem.



Recurrent Neural Network:

Recurrent Neural Network is a kind of artificial Intelligence in which the patterns in the data sequences such as text, handwriting, and spoken words are identified. It uses back propagation algorithm for training the model. It is known as back propagation through time as back propagation happen for every timestamp. Long Short-Term Memory Networks (LSTMs): These are the special kind of neural networks that are used for learning long-term dependencies, mainly in sequence prediction problems. It has a feedback connection that is itcan process the entire data sequence apart from single data points (images).

Responsibilities:

Sneha Latha Kusuma: working on the support vector machine to get better accuracy.

Bhavyasri Maddineni: working on random forest in order to work on various algorithms.

Shiny Sherly Katuru : working on Recurrent Neural Network to get better accuracy on different datasets.

Issues/Concerns:

- In order to get more accurate result the number of epochs has to be increased. The run time of the epochs is very high. So, the main concern is the time constraint.
- We have trained different models on the large datasets to get better accuracy.

References:

- **https://www.researchgate.net/publication/298808334_Handwritten_Text_Recognition_System_based_on_Neural_Network**
- **http://www.ijcsit.com/docs/Volume%207/vol7issue1/ijcsit2016070101.pdf**
- **http://cdn.iiit.ac.in/cdn/cvit.iiit.ac.in/images/ConferencePapers/2018/improving-cnn-rnn.pdf**
- **http://cs231n.stanford.edu/reports/2017/pdfs/810.pdf**
- **Handwritten Chinese Text Recognition Using Separable Multi-Dimensional Recurrent Neural Network | IEEE Conference Publication | IEEE Xplore**