

Problem Statement

The problem in hand seeks the classification of Arabic MNIST and Arabic Characters using Machine Learning and Deep Learning technique . This technique are backed by using advance computer vision approach. The dataset is collected from <http://datacenter.aucegypt.edu> and <https://www.kaggle.com/mloey1/ahcd1> database . This MHDBASE comprise of 10 classes of Arabic Numerical letter each class comprising of 7000 images and for ahcd1 compromise of 28 classes for Arabic Character with 13440 images for Training with 480 images per class for training and for Testing it compromise of 3360 images. The total images sums up to 70000 images for MHDBASE for Arabic Numeric and for Arabic Character it has 16800 images. Both Machine Learning and Deep Learning model have been used for the experiments . Model in Machine Learning have been used and experimented extensively followed by Deep Learning models. Then an Hybrid of both Machine Learning and Deep Learning model is used. This hybrid models using comprise of CNN model as an extractor followed by Machine Learning model as a linear classifier. Models like SVM , RANDOM FOREST, XGBOOST, ADABOOST , CATBOOST AND Logisitic Regression has been used as a second model in HYbrid models.

For stand alone experiments ,use of weak learnables and boosting techniques have been used. Model like XGBOOST, CATBOOST and ADABOOST have been used followed by Random Forest, SVM , Logistic Regression is used as model comparison. For deep learning techniques, CNN and Feed Forward Neural Network with hyperparameter tuning and various Optimization has been used .

Introduction

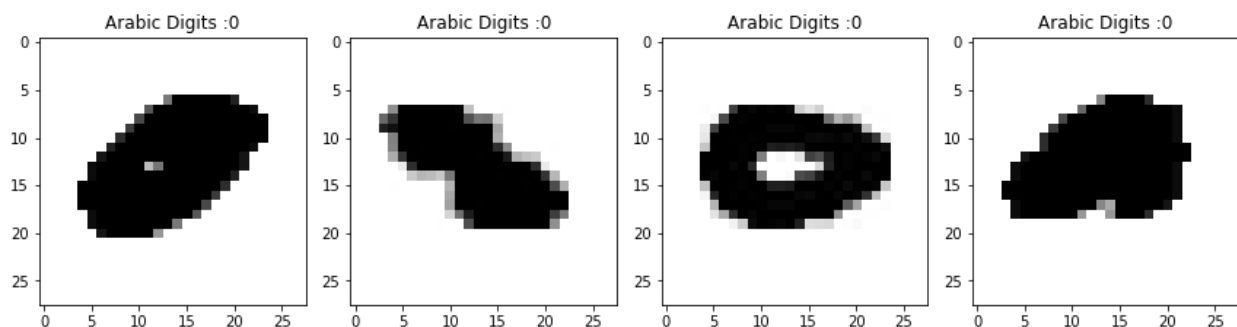
Machine Learning is the study of computer algorithms that improve consequently through experience.It is viewed as a subset of man-made brainpower. Machine Learning is actually the construction of a scientific model dependent on data points

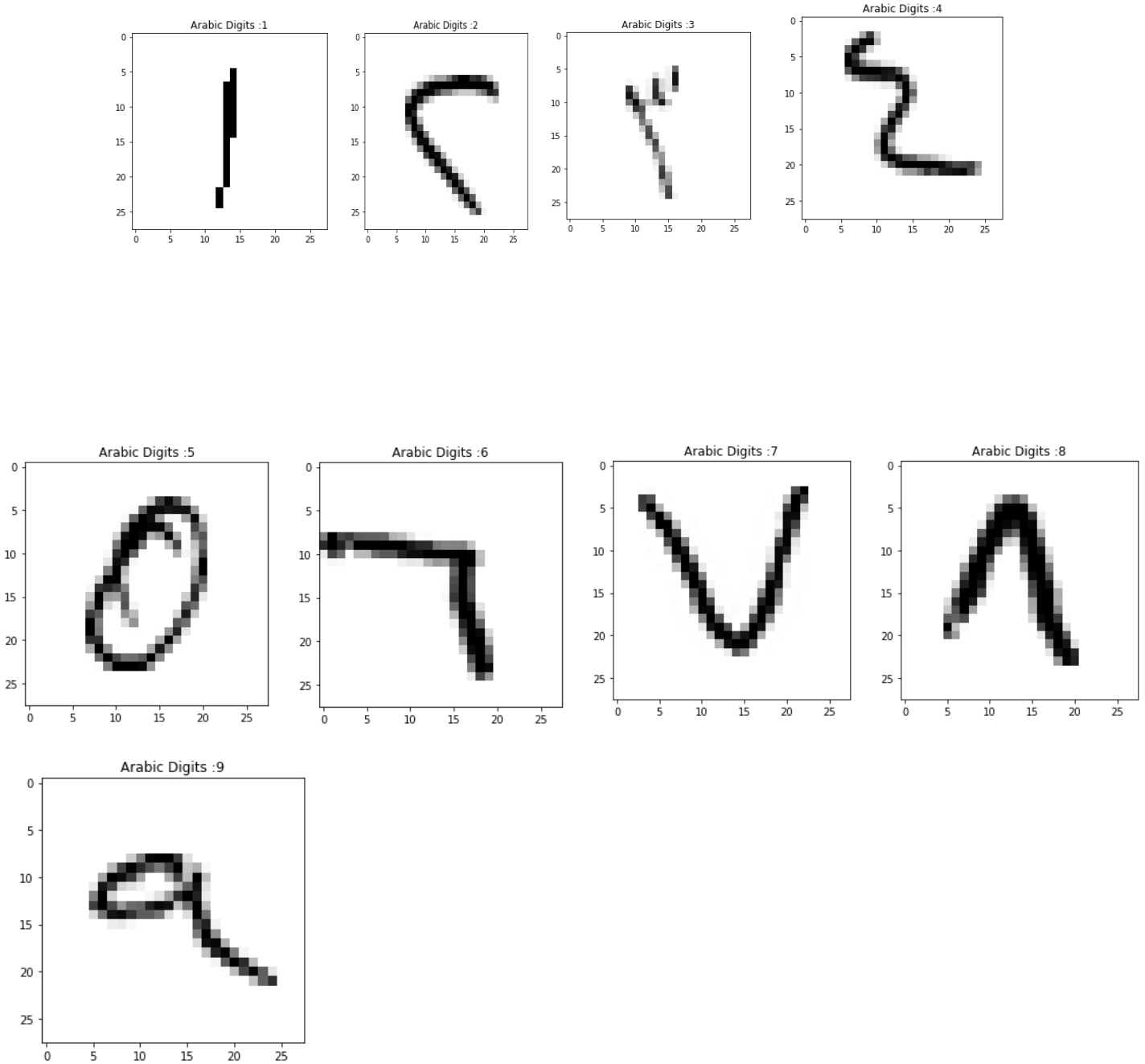
, known as "preparing information", so as to settle on expectations or choices without being expressly customized to do so. Machine Learning calculations are utilized in a wide assortment of uses, for example, email spam detection and Machine vision, where it is troublesome or infeasible to create customary calculations to play out the required assignments.

Machine Learning is firmly identified with computational statistics , which centers around making forecasts utilizing computational power. The investigation of scientific enhancement conveys techniques, hypothesis and application spaces to the field of AI. Data mining is a related field of study, concentrating on exploratory information examination through independent learning. In its application across business issues.

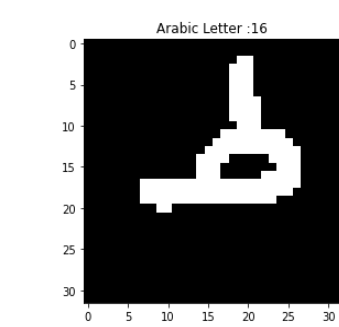
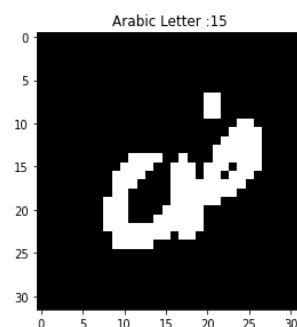
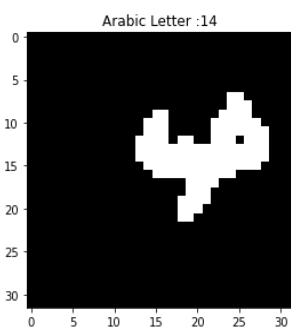
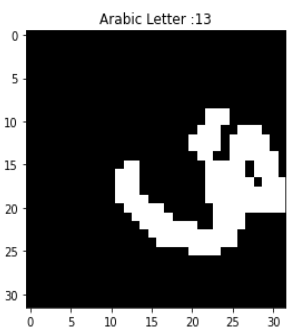
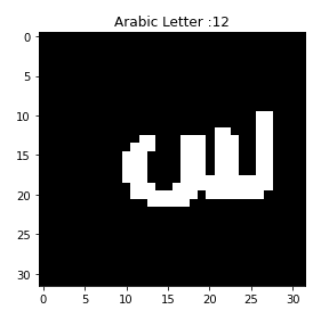
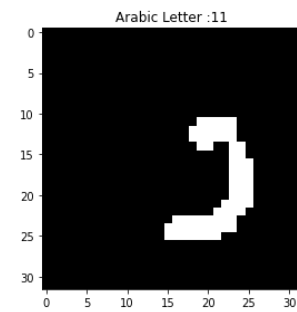
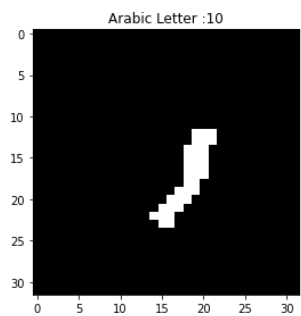
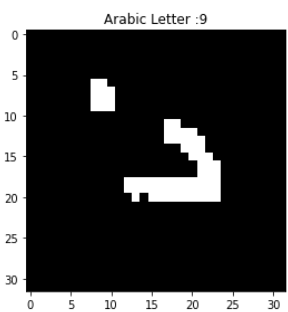
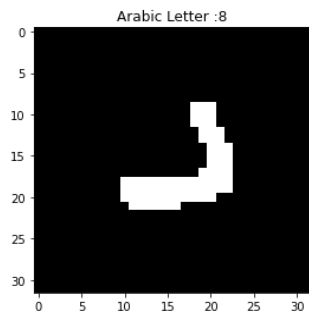
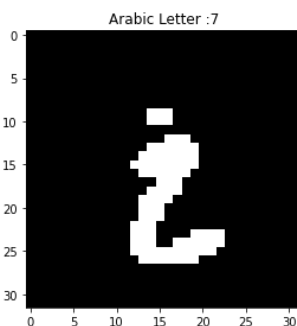
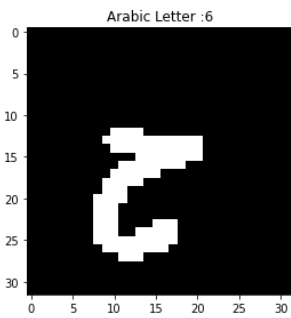
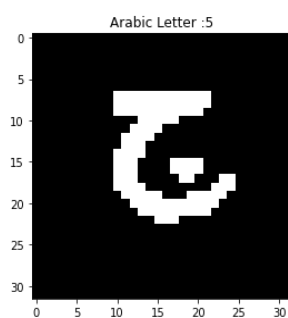
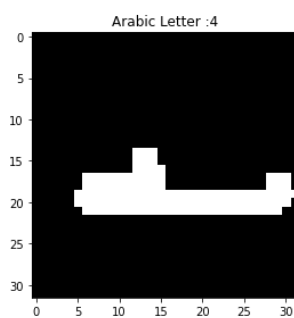
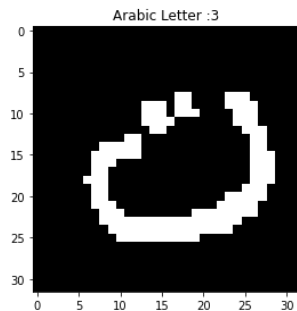
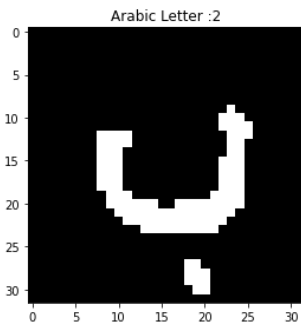
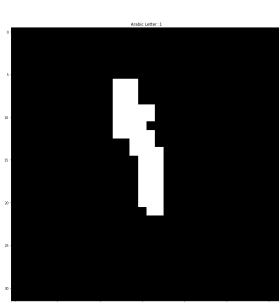
About the dataset

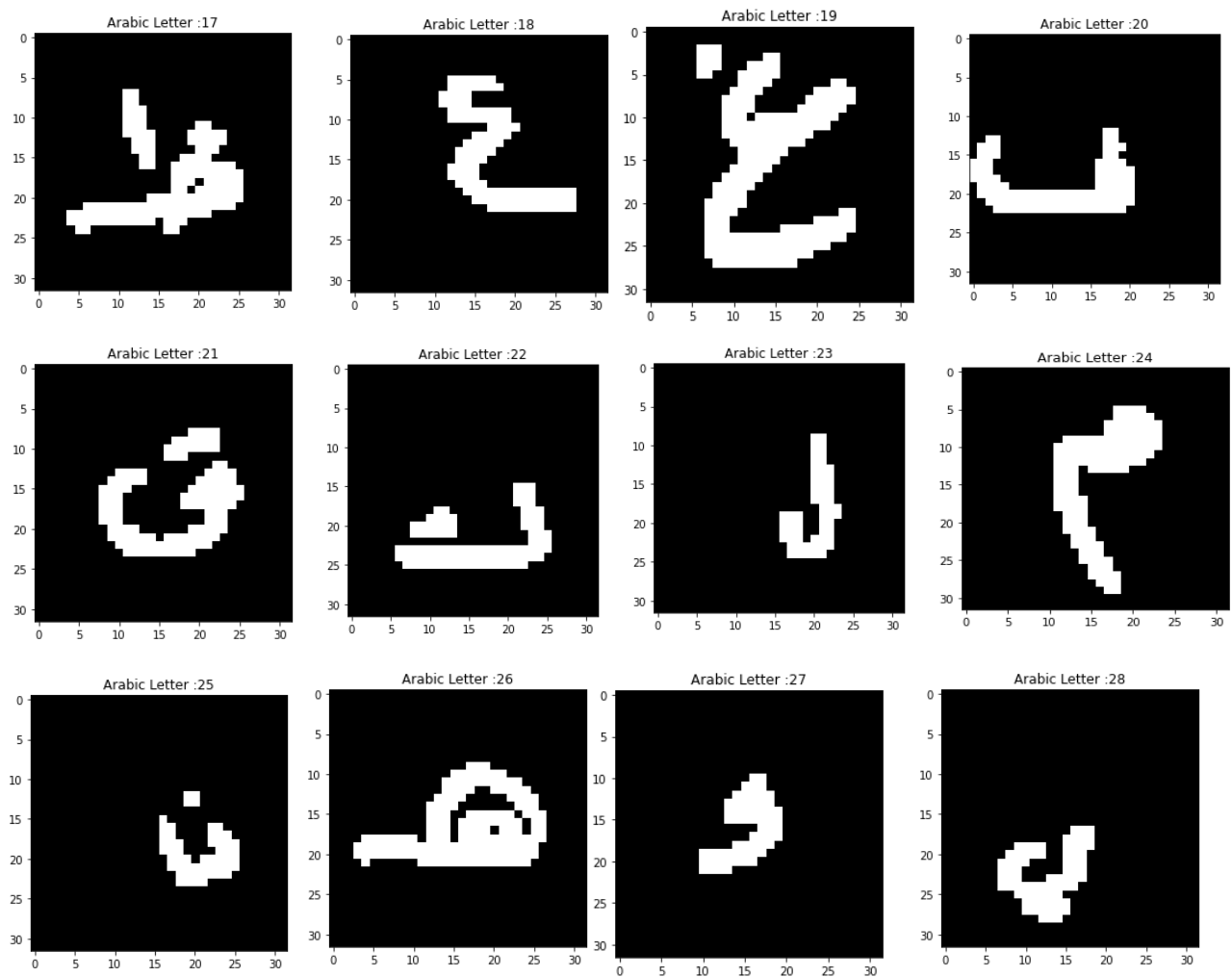
The dataset for the Arabic MNIST has been followed according to the MNIST dataset developed by *LeCun* et al. (1999) which comprises 60000 images for training and 10000 for testing with 6000 images per class. In same fashion , the images Arabic MNIST comprises 60000 images from digit 0 to digit 9 with each class having 6000 handwritten images .





The second dataset comprises 16800 characters written by 60 participants. The images were scanned at resolution 300 dpi. Each image is segmented automatically using the Matlab 2016a , which automatically coordinates every image. The database is partitioned into 13400 images as a training set with 480 images each class and a test set comprising 120 images per class , which sums to 3,360 images.





The above images are for 28 classes which resembles 28 characters in the arabic character

Set. The image resolution is 32x32 for every image in the class. The csv file for this character dataset came along with the image dataset from kaggle.

Data Processing and CSV conversion

The images were processed and converted using openCv and converted to grayscale so that images convert to a single filter from 3 filters. Row wise value will be extracted and will be extracted side by side to 784 columns. The label of

the given images were used as a target value and a csv file was thus generated. The kaggle dataset already has a CSV file with 1024 columns. These columns contain 0 for black value and 1 for white value in the image. The dataset was already made into two separate csv file for train and test.

Literature Review

[1] Uses the LeNet on Arabic MNIST. LeNet architecture was introduced by LeCun et al. in their 1998 paper, [*Gradient-Based Learning Applied to Document Recognition*](#). LeNet has a straightforward architecture and is small. Being small it is fast to train and also the performance has little tradeoff to accuracy. The paper was introduced mainly to combat OCR and character recognition. LeNet was officially designed for CPUs and no GPUs. The LeNet architecture is as follows INPUT => CONV => RELU => POOL => CONV => RELU => POOL => FC => RELU => FC. They have used MADBase image dataset for their suggested approach. Their model yielded 1% training and 12% testing errors. [2] uses a two stage system study and hence suggests a fast two stage classification which achieves high accuracy and less recognition time thereby decreasing latency. Two layer Neural Network has been implemented. ANNs are trained with the following numbers of hidden neurons $h=\{50,100,150,200,250,300\}$. The value with which h values the highest accuracy for validation set is used. The constant c for linear SVM is chosen from the given $C=\{0.1,1,10,100\}$. The one which yields the highest accuracy in the validation set was finalized. The same for the SVM RBF where the value for c and γ were determined from $\gamma=\{0.01,0.1,1,10\}$ and $c=10$ was kept constant. The two stage classification system yields 99.45 accuracy. [3] The paper uses a convolutional neural network for their model and the model yields 99.76% accuracy on validation set and the model comprises two layer CNN model. The parameters were empirically chosen and the epochs = 850 and batch size = 128 was optimally chosen. [4] The paper describes the comparison between latin and arabic letter classification and it does a review on 10 types of classifiers and accuracy. The paper states that the Arabic digit letter recognition is relatively easily than the latin character recognition. This is because the interclass difference in the latin classes is smaller than the arabic letter and variance in letter in latin is

larger. The paper reveals that the powerful classifiers show good performance on both latin and arabic but weak classifiers fail to perform better on Latin than on Arabic.[5] In this paper LVQ has been used for the first time as classifier. Three training algorithms were introduced in the paper mainly : OLVQ1, LVQ2 and LVQ3 and the best model out of the three is OLVQ1 is the best model . The model yielded 92.21 %, 88.24 % and 83.2 % accuracy . However, this showed a new approach but didn't have as good accuracy as Neural Network did. [7] Proposed methodology in this paper used BAMMLP with back propagation yielding 82% accuracy. The method is based on local image sampling of each character in the Arabic Character Set. Learning rate was 0.1 and momentum 0.25 and 35000 iterations were chosen as a value for the parameters. [8] This paper is very different from all the above experiments , this paper make utilization of hybrid model, where the model such as CNN act as a feature extractor then those feature extracted is fed into the SVM model as a linear classifier . This models are hybrid and take advantage of the model's pros with a little trade off with the model's cons. The model with CNN based-SVM with Dropout yields 2.09% error classification rate for 24 set classes and 5.83% for 66 HACDB set classes.

Proposed Methodology

Doing literature review , the machine models haven't been extensively used as Deep learning models, most of the papers which were implemented generally focused on feature ANN and CNN approach. Most of the advance ensemble methods like Categorical Boosting (CATBoost) and XGBoost weren't used. In the proposed methodology , we start by experimenting with the machine learning models like SVM linear, SVM RBF, Logistic Regression, Random Forest, Adaboost, XGBoost, CatBoost ,ANN, CNN, Transfer Learning and finally implementation of hybrid models of CNN+SVM , CNN XGBOOST, CNN+CATBOOST, CNN+LOGISTIC REGRESSION , CNN+ RANDOM FOREST .

All the models were trained with a ratio 70 % of data and tested with 30% of data. Two datasets where used , one of the dataset was used for the Arabic MNIST comprising 10 classes for 0-9 Numerical dataset and the other dataset which was


used was Arabic Character dataset which comprise of 28 classes resembling 28 Arabic Character.

1. XGBOOST

XGBoost stands for Extreme Gradient Boosting . It makes use of gradient boosting for decision trees thus increasing speed and performance. It is a supervised learning method and uses functional approximation for optimizing the loss function. It also achieves better accuracy by applying a regularization method.

To get a better accuracy, the model after t iterations tries to minimize the following objective function which comprise of loss function and regularization.

$$\mathcal{L}^{(t)} = \sum_{i=1}^n l(\overset{\substack{\text{Real value (label) known} \\ \text{from the training data-set}}}{y_i}, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t)$$



 Can be seen as $f(\mathbf{x} + \Delta\mathbf{x})$ where $\mathbf{x} = \hat{y}_i^{(t-1)}$

Three main form of boosting :

1. Gradient Boosting algorithm uses the gradient algorithm using learning rate.
2. Stochastic Gradient Boosting algorithm using sub sampling at the row and column per split levels.
3. Regularized Gradient Boosting uses L1 and L2 Regularization

Classification report for Arabic MNIST

XGboost accuracy : 1.00

Class	precision	recall	f1-score	support
0	1.0	1.0	1.0	2135
1	1.0	1.0	1.0	2109
2	1.0	1.0	1.0	2092
3	1.0	1.0	1.0	2121
4	1.0	1.0	1.0	2042
5	1.0	1.0	1.0	2120
6	1.0	1.0	1.0	2021
7	1.0	1.0	1.0	2134
8	1.0	1.0	1.0	2035
9	1.0	1.0	1.0	2111
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

Classification report Arabic Character dataset:

XG boost accuracy: 0.48549107142857145

Classification report

Class	precision	recall	f1-score	support
1	0.70	0.81	0.75	100
2	0.40	0.61	0.48	83
3	0.29	0.27	0.28	88
4	0.57	0.40	0.47	107
5	0.43	0.41	0.42	104
6	0.33	0.28	0.30	101
7	0.33	0.28	0.30	89
8	0.46	0.59	0.51	99
9	0.43	0.40	0.42	84
10	0.37	0.61	0.46	77
11	0.49	0.68	0.57	87
12	0.49	0.42	0.45	100
13	0.60	0.53	0.56	85
14	0.56	0.55	0.55	98
15	0.51	0.52	0.52	103
16	0.40	0.34	0.36	101
17	0.39	0.36	0.38	97
18	0.48	0.46	0.47	89
19	0.53	0.44	0.48	110
20	0.45	0.43	0.44	100
21	0.46	0.46	0.46	94
22	0.65	0.61	0.63	99
23	0.62	0.62	0.62	104
24	0.45	0.66	0.53	93
25	0.38	0.47	0.42	86
26	0.75	0.47	0.58	113
27	0.54	0.42	0.47	98
28	0.64	0.55	0.59	99
accuracy			0.49	2688
macro avg	0.49	0.49	0.48	2688
weighted avg	0.49	0.49	0.48	2688

Random Forest

A Random forest is a meta estimator that fits a number of different decision trees on various sub samples and the output is averaged and helps control overfitting. It is an ensemble learning method and used for regression , classification and other tasks that include constructing numerous decision trees at training time and outputting the class that is the mode of the prediction and hence classification ; and for regression,outputting the mean of the class.

Random forest uses the following mode of practise in particular :

1. Using the out of bag error , to estimate and reduce the generalization error.
2. Using permutation to find out the variable importance

Decision trees are a popular method of machine learning task. Tree learning thus comes off the shell method for data mining because of its invariant behaviour when it comes to scaling and other transformations methods for feature values which is robust to inclusion of feature values. The error generated by random forest is always lesser than the decision tree because of the out of bag error , also called the out of bag estimate. This estimating error method technique is a method of estimating the prediction error of random forests, boosted decision trees utilizing the bootstrap aggregating to subsample data samples required for training.

Parameter : max_depth=12, random_state=0, n_estimators=100

Classification report

Class	precision	recall	f1-score	support
0	0.99	0.98	0.99	2135
1	0.98	1.0	0.99	2109
2	0.99	0.99	0.99	2092
3	0.98	0.98	0.98	2121
4	0.99	0.99	0.99	2042
5	0.98	0.98	0.98	2120
6	0.99	0.99	0.99	2134
7	0.99	0.99	0.99	2035
8	0.99	0.98	0.99	2111
9	0.99	1.0	0.99	2101
accuracy			0.99	21000
Macro avg	0.99	0.99	0.99	21000
Weighted avg	0.99	0.99	0.99	21000

Accuracy = 0.9871428571428571

Classification report for Arabic Character dataset

Random forest 0.6335565476190477

Classification report

	Class	precision	recall	f1-score	support
	1	0.84	0.89	0.86	100
	2	0.57	0.78	0.66	83
	3	0.40	0.49	0.44	88
	4	0.63	0.50	0.55	107
	5	0.69	0.59	0.63	104
	6	0.49	0.61	0.55	101
	7	0.47	0.39	0.43	89
	8	0.55	0.78	0.65	99
	9	0.59	0.51	0.55	84
	10	0.52	0.81	0.63	77
	11	0.67	0.72	0.70	87
	12	0.67	0.66	0.67	100
	13	0.77	0.66	0.71	85
	14	0.63	0.63	0.63	98
	15	0.68	0.56	0.62	103
	16	0.61	0.55	0.58	101
	17	0.52	0.52	0.52	97
	18	0.58	0.58	0.58	89
	19	0.67	0.52	0.58	110
	20	0.56	0.57	0.57	100
	21	0.57	0.54	0.55	94
	22	0.89	0.73	0.80	99
	23	0.83	0.81	0.82	104
	24	0.68	0.85	0.75	93
	25	0.51	0.70	0.59	86
	26	0.89	0.63	0.74	113
	27	0.71	0.72	0.72	98
	28	0.84	0.48	0.62	99
	accuracy			0.63	2688
	macro avg	0.64	0.64	0.63	2688
	weighted avg	0.65	0.63	0.63	2688

CatBoost

CATBOOST stands for category boosting. It is especially powerful in two ways:

1. It delivers state of the art results without extensive data training typically required by other machine learning methods.
2. It provides a powerful descriptive method for out of box problems and business problems.

Catboost algorithm was developed by Yandex

Cat Boost uses gradient boosting technique and does not require conversion of data set to specific type format unlike other machine learning algorithms . This thus makes it more reliable and easy to implement algorithms.

Classification report for Arabic MNIST

Parameter : iterations=100 , depth=4, learning_rate=0.1

Class	precision	recall	f1-score	support
0	0.98	0.95	0.96	2135
1	0.94	0.98	0.0.96	2109
2	0.96	0.95	0.96	2092
3	0.93	0.95	0.94	2121
4	0.95	0.94	0.95	2042
5	0.96	0.95	0.95	2120
6	0.96	0.96	0.96	2134
7	0.96	0.97	0.97	2035

8	0.96	0.95	0.96	2111
9	0.94	0.94	0.94	2101
accuracy			0.95	21000
Macro avg	0.95	0.95	0.95	21000
Weighted avg	0.95	0.95	0.95	21000

Accuracy = 0.9547619047619048

Classification report for Arabic Character dataset

CatBoost 0.4642857142857143

Classification report

Class	precision	recall	f1-score	support
1	0.67	0.82	0.74	100
2	0.39	0.61	0.47	83
3	0.24	0.23	0.23	88
4	0.57	0.43	0.49	107
5	0.46	0.38	0.42	104
6	0.33	0.30	0.31	101
7	0.38	0.30	0.34	89
8	0.43	0.53	0.47	99
9	0.42	0.43	0.43	84
10	0.33	0.68	0.44	77
11	0.49	0.48	0.49	87
12	0.56	0.49	0.52	100
13	0.43	0.36	0.39	85
14	0.48	0.48	0.48	98
15	0.48	0.39	0.43	103
16	0.49	0.36	0.41	101
17	0.41	0.41	0.41	97
18	0.40	0.39	0.40	89
19	0.58	0.42	0.49	110
20	0.44	0.37	0.40	100
21	0.38	0.41	0.39	94

22	0.57	0.57	0.57	99
23	0.61	0.54	0.57	104
24	0.45	0.69	0.54	93
25	0.35	0.45	0.40	86
26	0.67	0.47	0.55	113
27	0.51	0.49	0.50	98
28	0.62	0.55	0.58	99
accuracy			0.46	2688
macro avg	0.47	0.47	0.46	2688
weighted avg	0.48	0.46	0.46	2688

Logistic Regression

Logistic Regression uses one versus all rest method . It uses L1, L2 and ElasticNet as regularization techniques. Logistic Regression calculates the probability of a particular set of data points belonging to either of those classes. The use of sigmoid of activation function and exponent makes it a probabilistic model as it strictly follows the probability rule of value always greater than 0 and less than equals to 1. The log-likelihood of the event is given by taking log of the equation. The log transformation changes the value from negative to positive transformation . Second , the log values are easy to interpret. Thus the cost function of the logistic regression is taken log. The presence of minus sign in the beginning of the function is to ensure that we are minimizing the negative of the likelihood instead of maximizing the value as gradient descent minimizes the error.

Classification report for Arabic MNIST

Class	precision	recall	f1-score	support
0	1.0	1.0	1.0	2135
1	1.0	1.0	1.0	2109
2	1.0	1.0	1.0	2092

3	1.0	1.0	1.0	2121
4	1.0	1.0	1.0	2042
5	1.0	1.0	1.0	2120
6	1.0	1.0	1.0	2021
7	1.0	1.0	1.0	2134
8	1.0	1.0	1.0	2035
9	1.0	1.0	1.0	2111
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

Accuracy = 0.9995714285714286

Classification report for Arabic Character Dataset:

Logistic Regression 0.6528728788329861

Classification report

Class	precision	recall	f1-score	support
1	0.87	0.94	0.90	119
2	0.68	0.78	0.73	120
3	0.45	0.48	0.47	120
4	0.61	0.50	0.55	120
5	0.52	0.61	0.56	120
6	0.53	0.53	0.53	120
7	0.53	0.41	0.46	120
8	0.54	0.68	0.60	120
9	0.54	0.57	0.56	120
10	0.60	0.76	0.67	120
11	0.65	0.59	0.62	120

12	0.72	0.72	0.72	120
13	0.79	0.78	0.78	120
14	0.61	0.57	0.59	120
15	0.63	0.61	0.62	120
16	0.61	0.66	0.63	120
17	0.61	0.54	0.58	120
18	0.54	0.57	0.56	120
19	0.62	0.58	0.60	120
20	0.56	0.58	0.57	120
21	0.60	0.56	0.58	120
22	0.91	0.75	0.82	120
23	0.93	0.80	0.86	120
24	0.80	0.84	0.82	120
25	0.58	0.68	0.63	120
26	0.81	0.82	0.81	120
27	0.75	0.71	0.73	120
28	0.83	0.69	0.75	120
accuracy			0.65	3359
macro avg	0.66	0.65	0.65	3359
weighted avg	0.66	0.65	0.65	3359

SVM

Support Vector Machine is a supervised machine learning algorithm which uses classification algorithms to group two different . The SVM takes these data points and outputs the hyperplane that separates the classes. The best hyperplane is the one that maximizes that margins between the classes .When the data sets aren't sepearable using the linear decision boundaries , we take a third dimension and take a slice of that space . This introduction of a new dimension is what comes under the kernel trick. Normally when the kernel is linear ,we get a linear classifier but when the kernel is non linear, we devise a new method which is a kernel trick, which without transforming the data at all ,we get a non - linear classifier. This we can get through by only changing the dot product of that space.

Classification report for ARABIC MNIST DATASET

Parameter: Kernel= RBF

Class	precision	recall	f1-score	support
0	1.0	1.0	1.0	2135
1	0.99	1.0	1.0	2109
2	1.0	0.99	0.99	2092
3	0.99	0.99	0.99	2121
4	0.99	1.0	1.0	2042
5	1.0	1.0	1.0	2120
6	1.0	1.0	1.0	2134
7	0.99	1.0	1.0	2035
8	1.0	0.99	1.0	2111
9	1.0	1.0	1.0	2101
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

Accuracy=0.9962857142857143

Classification report for Arabic Character dataset

Parameter : RBF

svm accuracy : 0.6528728788329861

Classification report

Class	precision	recall	f1-score	support
1	0.87	0.94	0.90	119
2	0.68	0.78	0.73	120
3	0.45	0.48	0.47	120
4	0.61	0.50	0.55	120
5	0.52	0.61	0.56	120
6	0.53	0.53	0.53	120
7	0.53	0.41	0.46	120
8	0.54	0.68	0.60	120
9	0.54	0.57	0.56	120
10	0.60	0.76	0.67	120
11	0.65	0.59	0.62	120
12	0.72	0.72	0.72	120
13	0.79	0.78	0.78	120
14	0.61	0.57	0.59	120
15	0.63	0.61	0.62	120
16	0.61	0.66	0.63	120
17	0.61	0.54	0.58	120
18	0.54	0.57	0.56	120
19	0.62	0.58	0.60	120
20	0.56	0.58	0.57	120
21	0.60	0.56	0.58	120
22	0.91	0.75	0.82	120
23	0.93	0.80	0.86	120
24	0.80	0.84	0.82	120
25	0.58	0.68	0.63	120
26	0.81	0.82	0.81	120
27	0.75	0.71	0.73	120
28	0.83	0.69	0.75	120
accuracy			0.65	3359
macro avg	0.66	0.65	0.65	3359
weighted avg	0.66	0.65	0.65	3359

Classification report for Arabic MNIST DATASET

Parameter: kernel='linear'

Class	precision	recall	f1-score	support
0	1.0	1.0	1.0	2135
1	1.0	1.0	1.0	2109
2	1.0	1.0	1.0	2092
3	1.0	1.0	1.0	2121
4	1.0	1.0	1.0	2042
5	1.0	1.0	1.0	2120
6	1.0	1.0	1.0	2134
7	1.0	1.0	1.0	2035
8	1.0	1.0	1.0	2111
9	1.0	1.0	1.0	2101
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

Accuracy = 0.9995714285714286

Classification report for Arabic Character Dataset

Parameter : Kernel ="Linear"

svm 0.423340279845192

Classification report

Class	precision	recall	f1-score	support
1	0.73	0.88	0.80	119
2	0.51	0.63	0.56	120
3	0.31	0.38	0.34	120
4	0.42	0.36	0.39	120
5	0.25	0.36	0.30	120
6	0.25	0.32	0.28	120
7	0.22	0.23	0.22	120
8	0.44	0.54	0.48	120
9	0.41	0.43	0.42	120
10	0.54	0.62	0.58	120
11	0.48	0.46	0.47	120
12	0.47	0.48	0.48	120
13	0.42	0.39	0.41	120
14	0.42	0.53	0.47	120
15	0.30	0.24	0.27	120
16	0.40	0.42	0.41	120
17	0.39	0.35	0.37	120
18	0.30	0.28	0.29	120
19	0.41	0.28	0.33	120
20	0.33	0.30	0.31	120
21	0.33	0.30	0.31	120
22	0.51	0.48	0.50	120
23	0.64	0.58	0.61	120
24	0.58	0.48	0.53	120
25	0.54	0.44	0.48	120
26	0.43	0.47	0.45	120
27	0.44	0.27	0.33	120
28	0.40	0.30	0.31	120
micro avg	0.42	0.43	0.42	3239
macro avg	0.41	0.41	0.41	3239
weighted avg	0.42	0.43	0.42	3239

Feed forward Neural Network

Architecture Design

Layer (type)	Output Shape	Param #
=====		
dense_30 (Dense)	(None, 512)	401920
=====		
dense_31 (Dense)	(None, 512)	262656
=====		
dense_32 (Dense)	(None, 10)	5130
=====		

Total params: 669,706

Trainable params: 669,706

Non-trainable params: 0

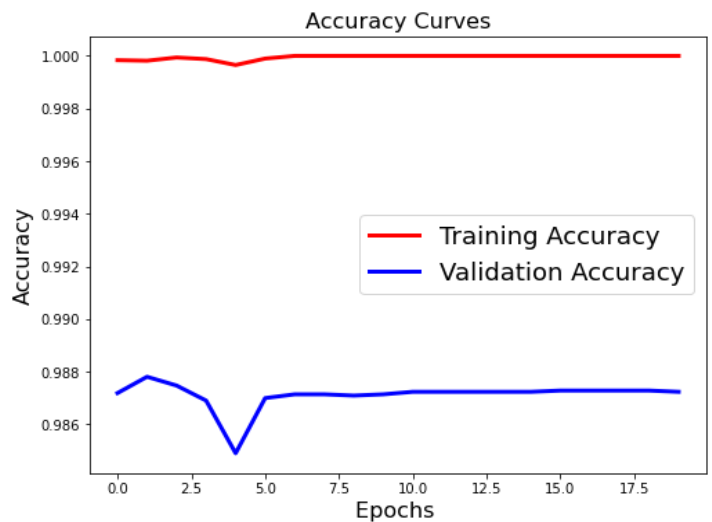
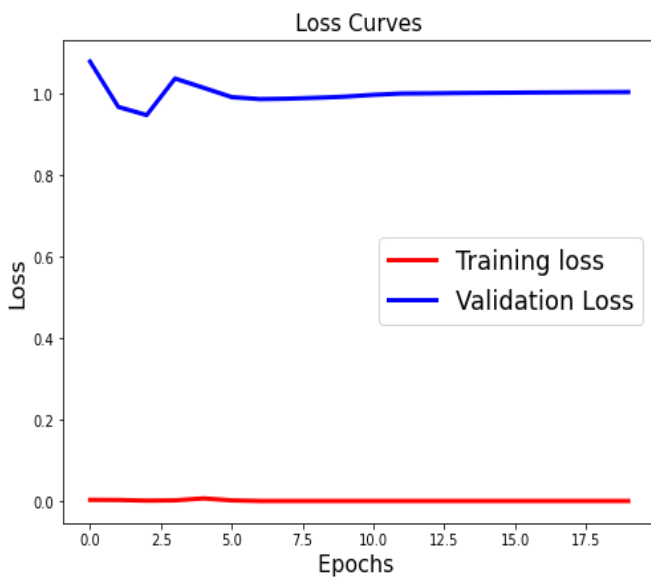
optimizer= 'adam'

CLASSIFICATION REPORT FOR ARABIC MNIST DATASET

Class	precision	recall	f1-score	support
0	0.98	0.98	0.98	2135
1	0.98	0.99	0.99	2109
2	0.98	0.99	0.99	2092

3	0.98	0.99	0.99	2121
4	0.99	0.98	0.99	2042
5	0.98	0.98	0.98	2120
6	0.99	0.99	0.99	2134
7	1.0	0.99	0.99	2035
8	0.99	0.99	0.99	2111
9	0.99	0.99	0.99	2101
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

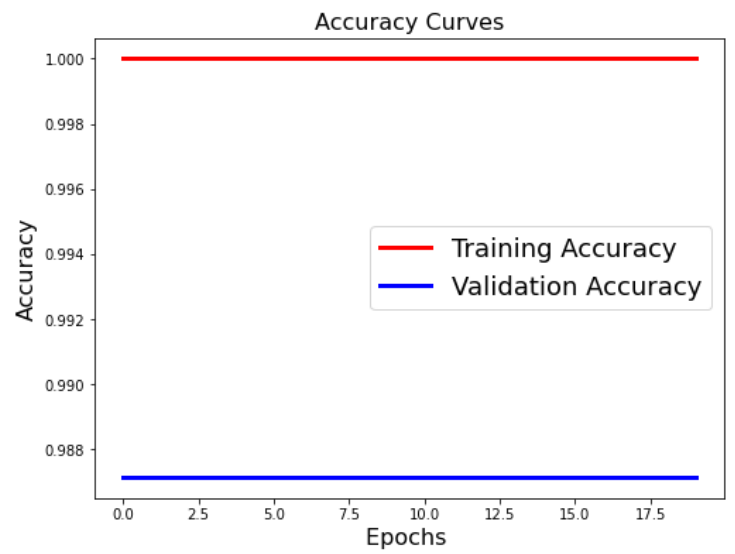
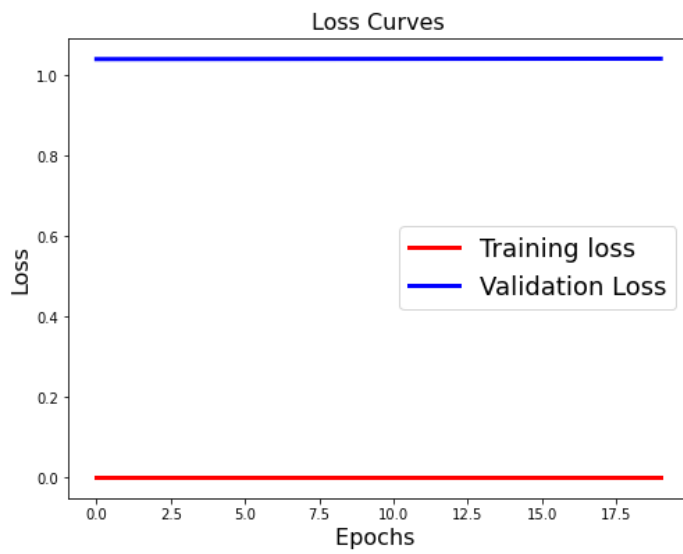
NN Prediction accuracy : 0.9871428571428571



Optimizer = RMSprop

Class	precision	recall	f1-score	support
0	0.98	0.98	0.98	2135
1	0.98	0.99	0.99	2109
2	0.98	0.99	0.99	2092
3	0.98	0.99	0.99	2121
4	0.99	0.98	0.99	2042
5	0.98	0.98	0.98	2120
6	0.99	0.99	0.99	2134
7	1.0	0.99	0.99	2035
8	0.99	0.99	0.99	2111
9	0.99	0.99	0.99	2101
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

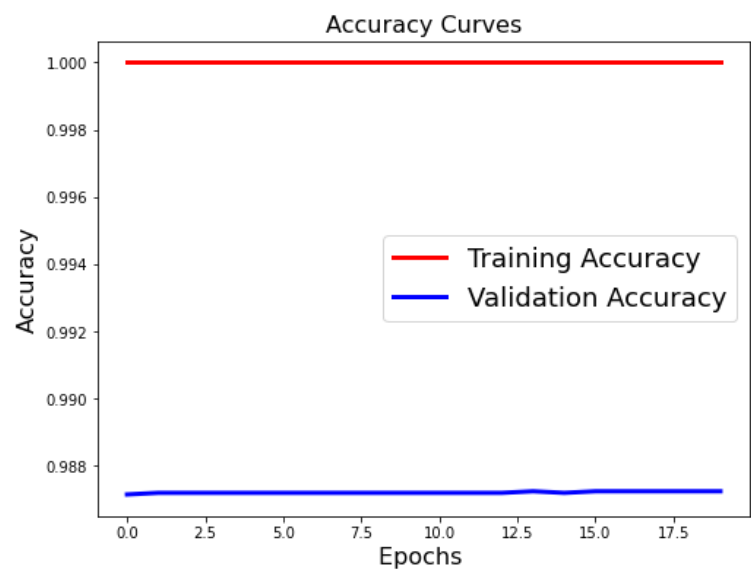
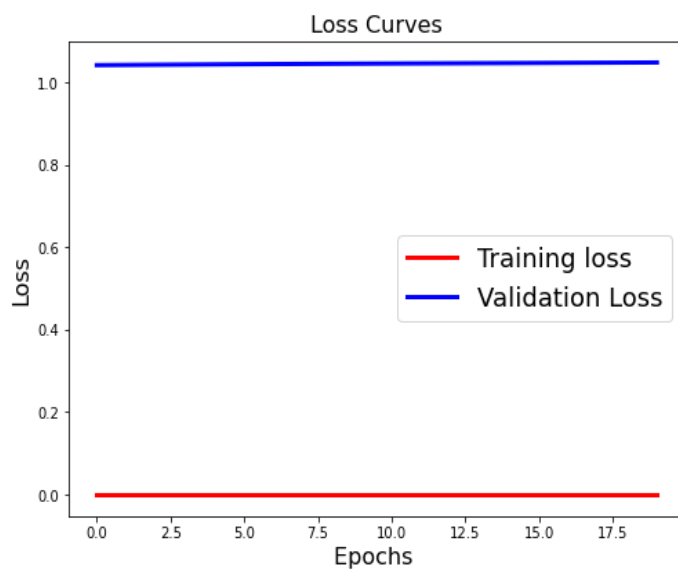
NN Prediction accuracy= 0.9871428571428571



optimizer='adagrad'

Class	precision	recall	f1-score	support
0	0.98	0.98	0.98	2135
1	0.98	0.99	0.99	2109
2	0.98	0.99	0.99	2092
3	0.98	0.99	0.99	2121
4	0.99	0.98	0.99	2042
5	0.98	0.98	0.98	2120
6	0.99	0.99	0.99	2134
7	1.0	0.99	0.99	2035

8	0.99	0.99	0.99	2111
9	0.99	0.99	0.99	2101
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

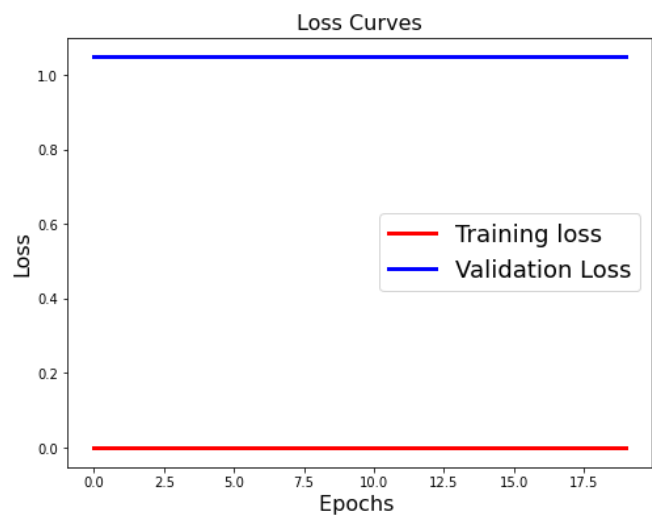
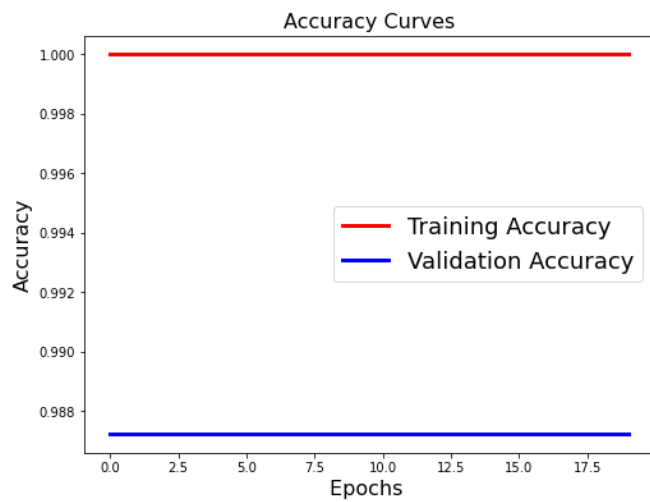


NN Prediction accuracy = 0.9872380952380952

optimizer='sgd'

Class	precision	recall	f1-score	support
0	0.98	0.98	0.98	2135

1	0.98	0.99	0.99	2109
2	0.98	0.99	0.99	2092
3	0.98	0.99	0.99	2121
4	0.99	0.98	0.99	2042
5	0.98	0.98	0.98	2120
6	0.99	0.99	0.99	2134
7	1.0	0.99	0.99	2035
8	0.99	0.99	0.99	2111
9	0.99	0.99	0.99	2101
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000



NN Prediction accuracy = 0.9872380952380952

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

Optimizer ='adagrad'

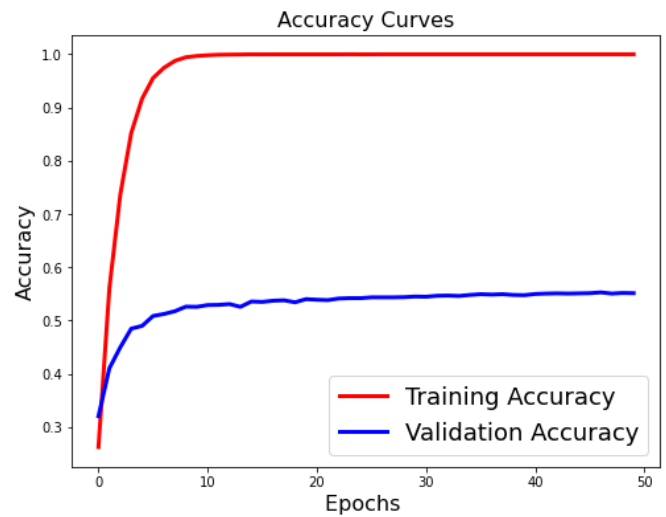
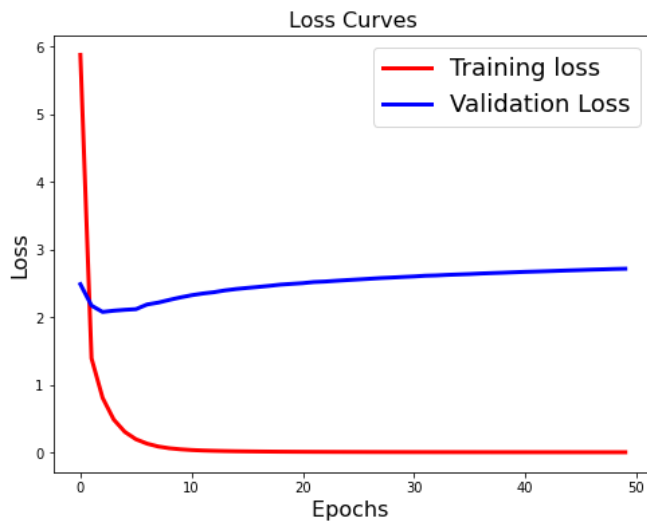
NN Prediction accuracy : 0.5375744047619048

Classification report

Class	precision	recall	f1-score	support
1	0.95	0.81	0.88	100
2	0.76	0.75	0.75	83
3	0.41	0.39	0.40	88
4	0.49	0.36	0.42	107
5	0.55	0.44	0.49	104
6	0.39	0.38	0.38	101
7	0.42	0.35	0.38	89
8	0.65	0.71	0.68	99
9	0.58	0.55	0.56	84
10	0.64	0.69	0.66	77
11	0.67	0.60	0.63	87
12	0.60	0.50	0.54	100
13	0.59	0.47	0.52	85
14	0.49	0.48	0.48	98
15	0.47	0.50	0.49	103
16	0.58	0.54	0.56	101
17	0.51	0.45	0.48	97
18	0.37	0.46	0.41	89
19	0.49	0.37	0.42	110
20	0.41	0.41	0.41	100
21	0.42	0.41	0.42	94
22	0.65	0.63	0.64	99
23	0.70	0.73	0.72	104
24	0.83	0.80	0.81	93
25	0.45	0.49	0.47	86
26	0.76	0.61	0.68	113
27	0.68	0.73	0.71	98
28	0.55	0.48	0.51	99

micro avg	0.57	0.54	0.55	2688
macro avg	0.57	0.54	0.55	2688
weighted avg	0.57	0.54	0.55	2688
samples avg	0.54	0.54	0.54	2688

Text(0.5, 1.0, 'Accuracy Curves')



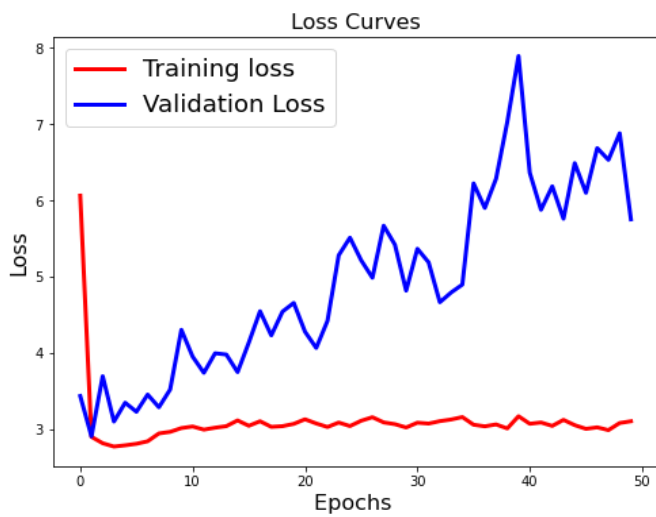
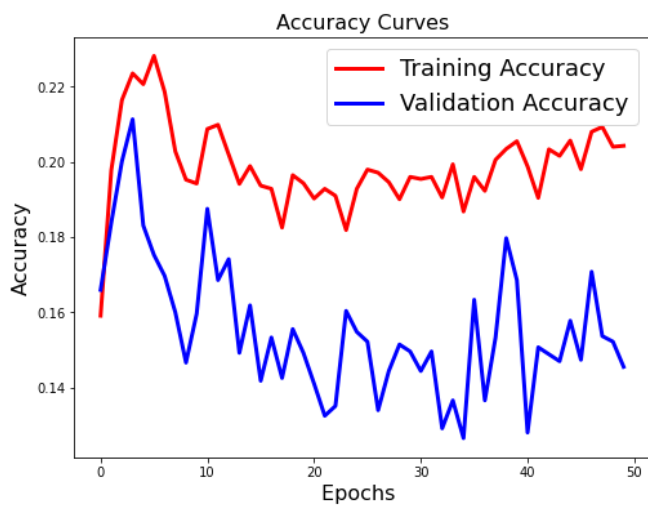
Optimizer='RMSProp'

NN Prediction accuracy : 0.11235119047619048

Classification report

Class	precision	recall	f1-score	support
1	0.92	0.85	0.89	100
2	0.80	0.19	0.31	83
3	0.00	0.00	0.00	88
4	0.56	0.05	0.09	107
5	0.00	0.00	0.00	104
6	0.00	0.00	0.00	101
7	0.33	0.01	0.02	89
8	0.87	0.39	0.54	99
9	0.38	0.04	0.07	84
10	0.63	0.47	0.54	77
11	0.58	0.08	0.14	87
12	0.00	0.00	0.00	100
13	0.00	0.00	0.00	85

14	0.00	0.00	0.00	98
15	0.00	0.00	0.00	103
16	0.00	0.00	0.00	101
17	0.00	0.00	0.00	97
18	0.00	0.00	0.00	89
19	0.60	0.08	0.14	110
20	0.00	0.00	0.00	100
21	0.00	0.00	0.00	94
22	0.00	0.00	0.00	99
23	0.77	0.52	0.62	104
24	0.90	0.39	0.54	93
25	0.67	0.02	0.04	86
26	0.00	0.00	0.00	113
27	1.00	0.09	0.17	98
28	0.00	0.00	0.00	99
micro avg	0.79	0.11	0.20	2688
macro avg	0.32	0.11	0.15	2688
weighted avg	0.32	0.11	0.14	2688
samples avg	0.11	0.11	0.11	2688

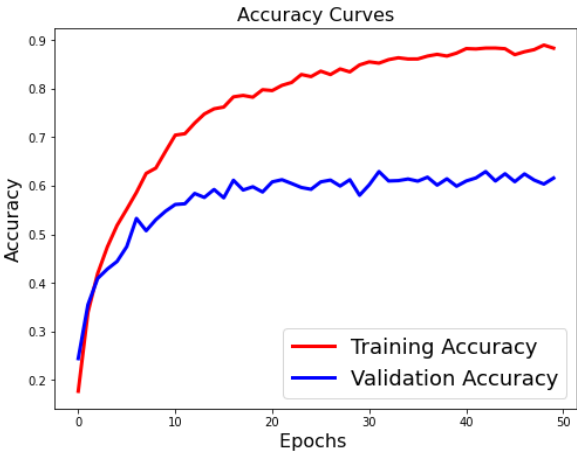
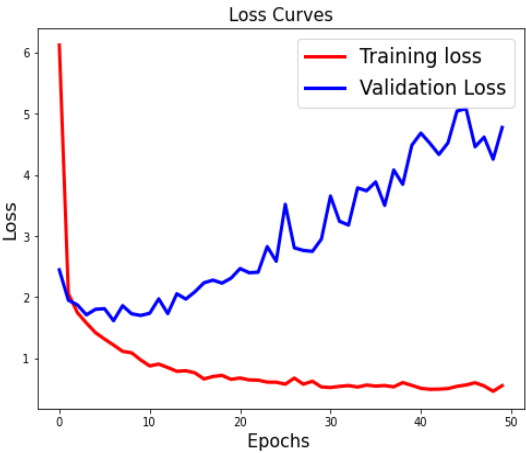


Optimizer = 'adam'

NN Prediction accuracy: 0.6101190476190477

Classification report

Class	precision	recall	f1-score	support
1	0.96	0.90	0.93	100
2	0.77	0.78	0.78	83
3	0.56	0.52	0.54	88
4	0.66	0.45	0.53	107
5	0.66	0.30	0.41	104
6	0.49	0.52	0.50	101
7	0.27	0.42	0.33	89
8	0.76	0.77	0.76	99
9	0.70	0.63	0.66	84
10	0.66	0.84	0.74	77
11	0.75	0.63	0.69	87
12	0.65	0.73	0.69	100
13	0.48	0.51	0.49	85
14	0.72	0.81	0.76	98
15	0.71	0.64	0.67	103
16	0.49	0.31	0.38	101
17	0.43	0.64	0.51	97
18	0.62	0.51	0.56	89
19	0.74	0.58	0.65	110
20	0.51	0.52	0.52	100
21	0.38	0.39	0.39	94
22	0.69	0.67	0.68	99
23	0.84	0.83	0.83	104
24	0.84	0.85	0.84	93
25	0.66	0.52	0.58	86
26	0.81	0.62	0.70	113
27	0.76	0.70	0.73	98
28	0.77	0.55	0.64	99
micro avg	0.64	0.61	0.63	2688
macro avg	0.66	0.61	0.63	2688
weighted avg	0.66	0.61	0.63	2688
samples avg	0.61	0.61	0.61	2688

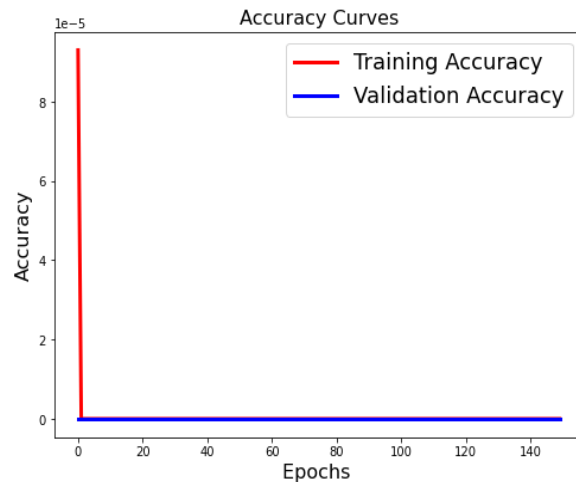
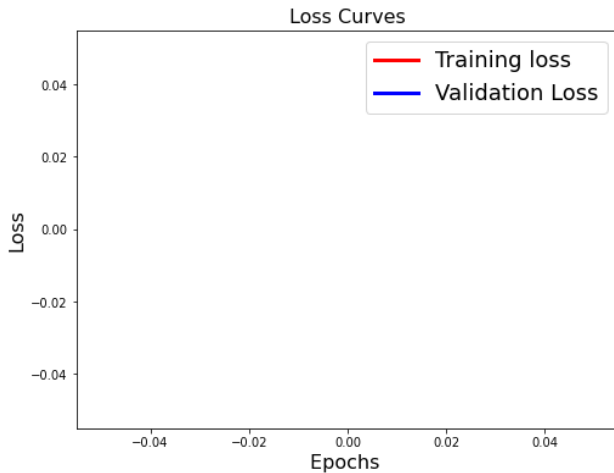


Optimizer = SGD

NN Prediction accuracy : 0.0 %

Classification report

	Class	precision	recall	f1-score	support
	1	0.00	0.00	0.00	100
	2	0.00	0.00	0.00	83
	3	0.00	0.00	0.00	88
	4	0.00	0.00	0.00	107
	5	0.00	0.00	0.00	104
	6	0.00	0.00	0.00	101
	7	0.00	0.00	0.00	89
	8	0.00	0.00	0.00	99
	9	0.00	0.00	0.00	84
	10	0.00	0.00	0.00	77
	11	0.00	0.00	0.00	87
	12	0.00	0.00	0.00	100
	13	0.00	0.00	0.00	85
	14	0.00	0.00	0.00	98
	15	0.00	0.00	0.00	103
	16	0.00	0.00	0.00	101
	17	0.00	0.00	0.00	97
	18	0.00	0.00	0.00	89
	19	0.00	0.00	0.00	110
	20	0.00	0.00	0.00	100
	21	0.00	0.00	0.00	94
	22	0.00	0.00	0.00	99
	23	0.00	0.00	0.00	104
	24	0.00	0.00	0.00	93
	25	0.00	0.00	0.00	86
	26	0.00	0.00	0.00	113
	27	0.00	0.00	0.00	98
	28	0.00	0.00	0.00	99
	micro avg	0.00	0.00	0.00	2688
	macro avg	0.00	0.00	0.00	2688
	weighted avg	0.00	0.00	0.00	2688
	samples avg	0.00	0.00	0.00	2688



CNN

CNN stands for Convolution Neural Network and is the main categories for image recognitions, classifications , Object Detection , Sentimental Analysis, Facial Recognition , Feature Extractor , etc .

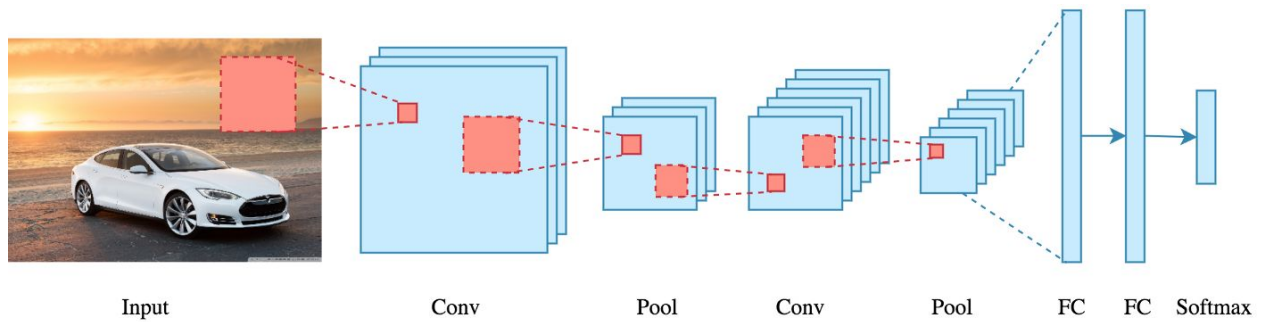
Convolutional Neural Networks (CNN) are everywhere. It's far arguably the most popular deep studying architecture. The current surge of hobby in deep learning is due to the substantial popularity and effectiveness of convnets.

The interest in CNN began with AlexNet in 2012 and it has grown exponentially ever given that in just three years, researchers progressed from eight layer AlexNet to 152 layer ResNet.

CNN is now the go-to version for any image detection task. In terms of accuracy , it is also efficaciously carried out to recommender structures, natural language processing and object detection . The main gain of CNN compared to its predecessors is that it routinely detects the vital features without any human supervision. For instance, given many photos of cats and puppies it learns one-of-a-kind capabilities for every class through itself.

Architecture:

All CNN models follow a comparable design, as appeared in the figure beneath.



Convolution:

The primary structure square of CNN is the convolutional layer. Convolution is a scientific activity to blend two arrangements of data. For our situation the convolution is applied on the info information utilizing a convolution channel to create a feature map.

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

Input

1	0	1
0	1	0
1	0	1

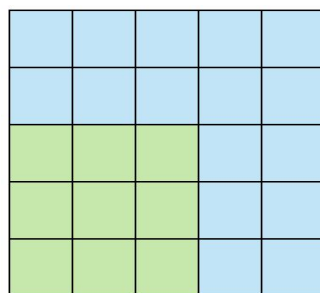
Filter / Kernel

Non-linearity:

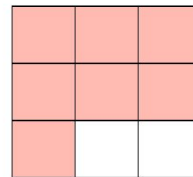
For any sort of neural system to be ground-breaking, it needs to contain non-linearity. Both the ANN and autoencoder we saw before accomplished this by passing the weighted entirety of its contributions through an actuation capacity, and CNN is the same. We again pass the aftereffect of the convolution activity through relu enactment work. So the qualities in the last element maps are not really the aggregates, however the relu work concerned them. We have precluded this in the figures above for straightforwardness. However, remember that any kind of convolution includes a relu activity, without that the system won't accomplish its actual potential.

Stride and Padding:

Step determines the amount we move the convolution channel at each progression. As a matter of course the worth is 1, as should be obvious in the figure underneath.



Stride 1

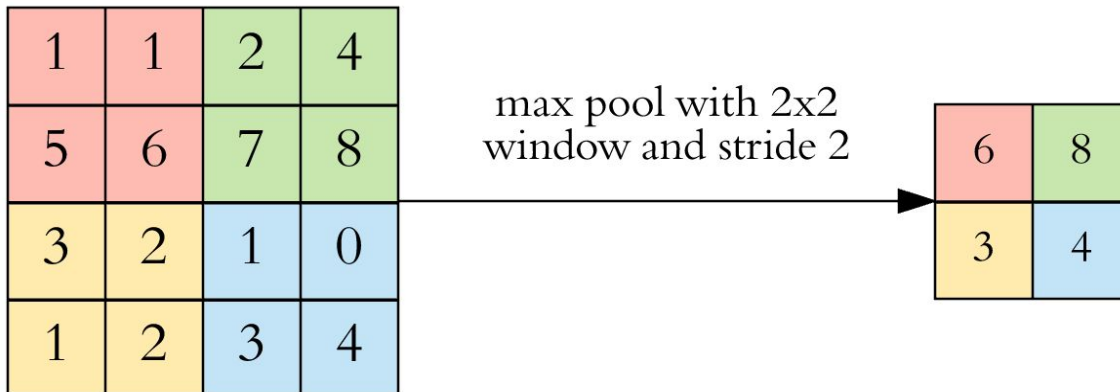


Feature Map

Pooling:

After a convolution activity we for the most part perform pooling to lessen the dimensionality. This empowers us to lessen the quantity of parameters, which both abbreviates the preparation time and battles overfitting. Pooling layers downsample each element map freely, decreasing the stature and width, keeping the profundity flawless.

The most well-known sort of pooling is max pooling which just takes the maximum incentive in the pooling window. In opposition to the convolution activity, pooling has no parameters. It slides a window over its info, and just takes the maximum incentive in the window. Like a convolution, we indicate the window size and step.



Architecture

Layer (type)	Output Shape	Param #
=====		
conv2d_7 (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d_7 (MaxPooling2)	(None, 13, 13, 32)	0
batch_normalization_7 (Batch Normalization)	(None, 13, 13, 32)	128
conv2d_8 (Conv2D)	(None, 11, 11, 64)	18496
max_pooling2d_8 (MaxPooling2)	(None, 5, 5, 64)	0
batch_normalization_8 (Batch Normalization)	(None, 5, 5, 64)	256
conv2d_9 (Conv2D)	(None, 3, 3, 128)	73856
max_pooling2d_9 (MaxPooling2)	(None, 1, 1, 128)	0

batch_normalization_9 (Batch Normalization) **512**

flatten_3 (Flatten) **(None, 128)** **0**

dense_5 (Dense) **(None, 256)** **33024**

dense_6 (Dense) **(None, 10)** **2570**

Total params: 129,162

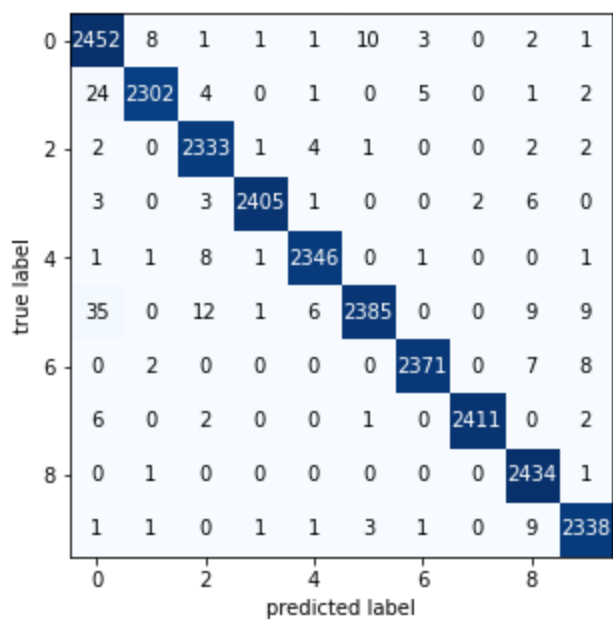
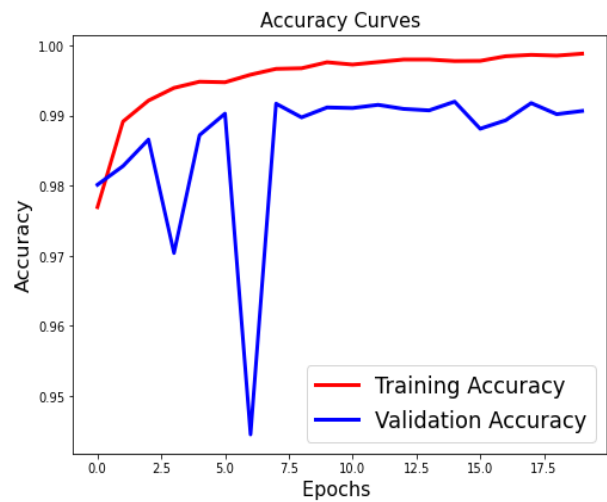
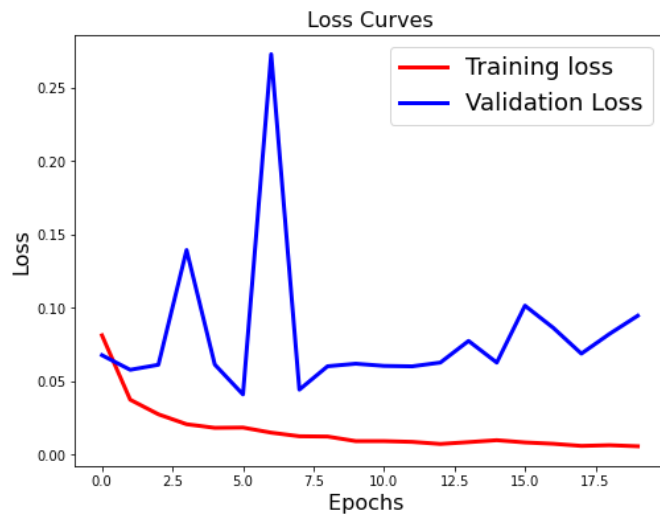
Trainable params: 128,714

Non-trainable params: 448

Class	precision	recall	f1-score	support
0	0.97	0.99	0.98	2479
1	0.99	0.98	0.99	2339
2	0.99	0.99	0.99	2345
3	1.0	0.99	1.0	2420
4	0.99	0.99	0.99	2359
5	0.99	0.97	0.98	2457
6	1.0	0.99	0.99	2388
7	1.0	1.0	1.0	2422
8	0.99	1.00	0.99	2436

9	0.99	0.99	0.99	2355
accuracy			1.0	21000
Macro avg	1.0	1.0	1.0	21000
Weighted avg	1.0	1.0	1.0	21000

CNN Accuracy : 0.990625



Transfer Learning

Class	precision	recall	f1-score	support
0	0.99	0.99	0.99	2452
1	0.99	0.99	0.99	2407
2	0.99	0.99	0.99	2400
3	1.0	1.0	1.0	2351
4	0.99	1.0	0.99	2415
5	1.0	1.0	1.0	2356
6	1.0	1.0	1.0	2447
7	1.0	1.0	1.0	2359
8	1.0	1.0	1.0	2440
9	0.99	1.0	0.99	2373
accuracy	1.0	1.0	1.0	11975
Micro avg	1.0	1.0	1.0	11975
Macro avg	1.0	1.0	1.0	11975
Weighted avg	1.0	1.0	1.0	11975

CNN Transfer Learning Accuracy : 0.9967432150313152

Classification report for Arabic Character Dataset

Architecture

Model: "sequential_1"

Layer (type)	Output Shape	Param #
=====		
conv2d_1 (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d_1 (MaxPooling2D)	(None, 13, 13, 32)	0
batch_normalization_1 (Batch Normalization)	(None, 13, 13, 32)	128
conv2d_2 (Conv2D)	(None, 11, 11, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 5, 5, 64)	0
batch_normalization_2 (Batch Normalization)	(None, 5, 5, 64)	256
conv2d_3 (Conv2D)	(None, 3, 3, 128)	73856
max_pooling2d_3 (MaxPooling2D)	(None, 1, 1, 128)	0
batch_normalization_3 (Batch Normalization)	(None, 1, 1, 128)	512
flatten_1 (Flatten)	(None, 128)	0
dense_1 (Dense)	(None, 256)	33024

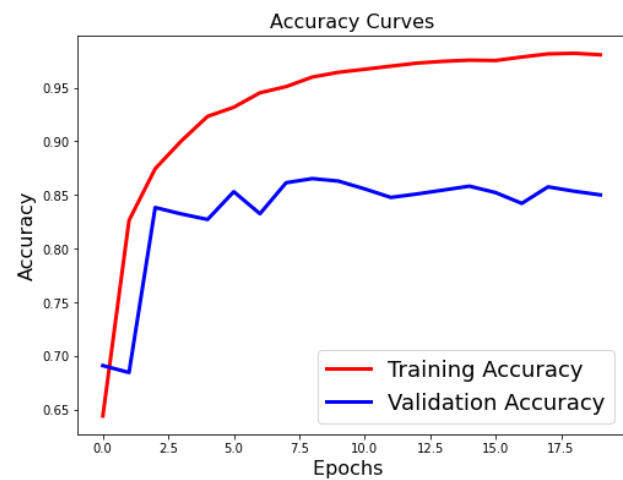
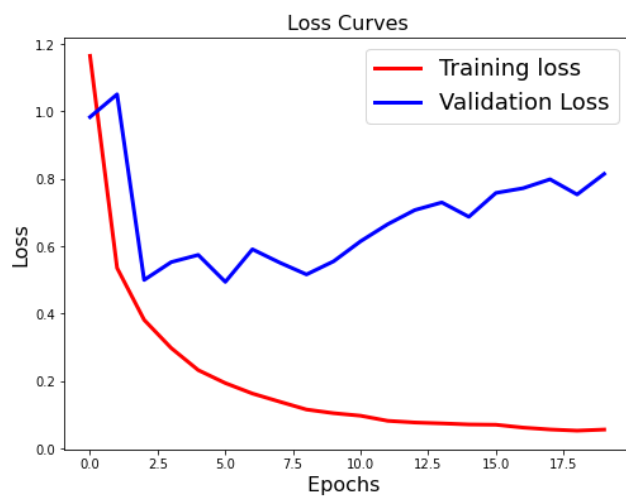
dense_2 (Dense) (None, 28) 7196
=====

Total params: 133,788
Trainable params: 133,340
Non-trainable params: 448

OPTIMIZER = RMSPROP()
NN Prediction accuracy 0.8482123742239349
Classification report

	precision	recall	f1-score	support
0	0.94	0.97	0.96	157
1	0.92	0.89	0.91	163
2	0.92	0.77	0.84	175
3	0.88	0.75	0.81	186
4	0.90	0.85	0.87	169
5	0.84	0.90	0.87	188
6	0.93	0.78	0.85	157
7	0.74	0.94	0.83	174
8	0.90	0.72	0.80	166
9	0.84	0.91	0.88	162
10	0.89	0.84	0.87	168
11	0.72	0.92	0.80	170
12	0.93	0.83	0.88	162
13	0.84	0.77	0.80	155
14	0.75	0.88	0.81	189
15	0.78	0.95	0.86	184
16	0.95	0.75	0.84	166
17	0.75	0.89	0.81	160
18	0.92	0.84	0.88	164
19	0.76	0.77	0.77	160
20	0.88	0.79	0.83	182
21	0.88	0.78	0.83	158
22	0.91	0.96	0.93	160
23	0.92	0.92	0.92	171
24	0.80	0.85	0.82	157
25	0.86	0.91	0.88	171
26	0.94	0.88	0.90	152

	27	0.91	0.72	0.80	145
micro avg		0.86	0.85	0.85	4671
macro avg		0.87	0.85	0.85	4671
weighted avg		0.86	0.85	0.85	4671
samples avg		0.85	0.85	0.85	4671



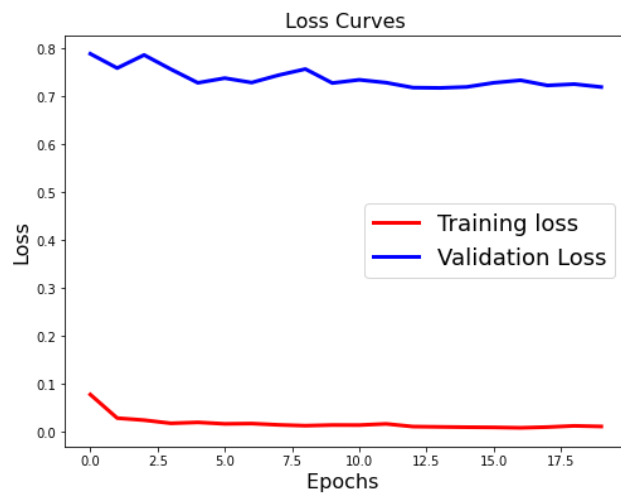
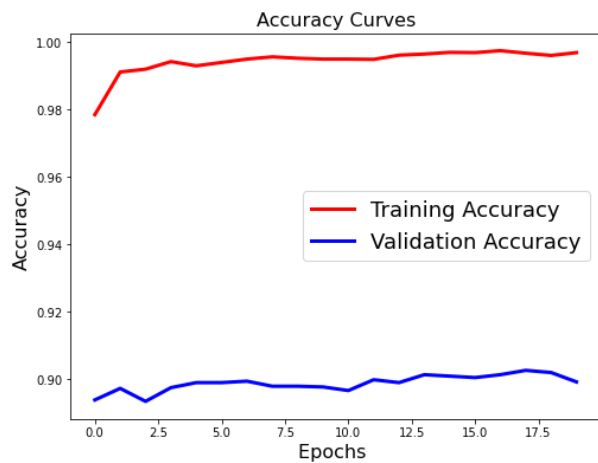
OPTIMIZER ='ADAGRAD'

NN Prediction 0.899165061014772

Classification report

CLASS	precision	recall	f1-score	support
0	0.97	0.99	0.98	157
1	0.95	0.96	0.95	163
2	0.90	0.91	0.90	175
3	0.90	0.88	0.89	186
4	0.91	0.90	0.90	169
5	0.87	0.91	0.89	188
6	0.96	0.85	0.91	157

7	0.88	0.94	0.91	174
8	0.88	0.81	0.85	166
9	0.87	0.90	0.89	162
10	0.85	0.85	0.85	168
11	0.88	0.93	0.91	170
12	0.91	0.91	0.91	162
13	0.89	0.90	0.90	155
14	0.92	0.89	0.90	189
15	0.90	0.90	0.90	184
16	0.90	0.89	0.90	166
17	0.83	0.91	0.87	160
18	0.94	0.92	0.93	164
19	0.82	0.79	0.81	160
20	0.93	0.80	0.86	182
21	0.93	0.88	0.91	158
22	0.94	0.96	0.95	160
23	0.94	0.94	0.94	171
24	0.83	0.90	0.87	157
25	0.93	0.92	0.93	171
26	0.91	0.93	0.92	152
27	0.94	0.91	0.92	145
micro avg	0.90	0.90	0.90	4671
macro avg	0.90	0.90	0.90	4671
weighted avg	0.90	0.90	0.90	4671
samples avg	0.90	0.90	0.90	4671

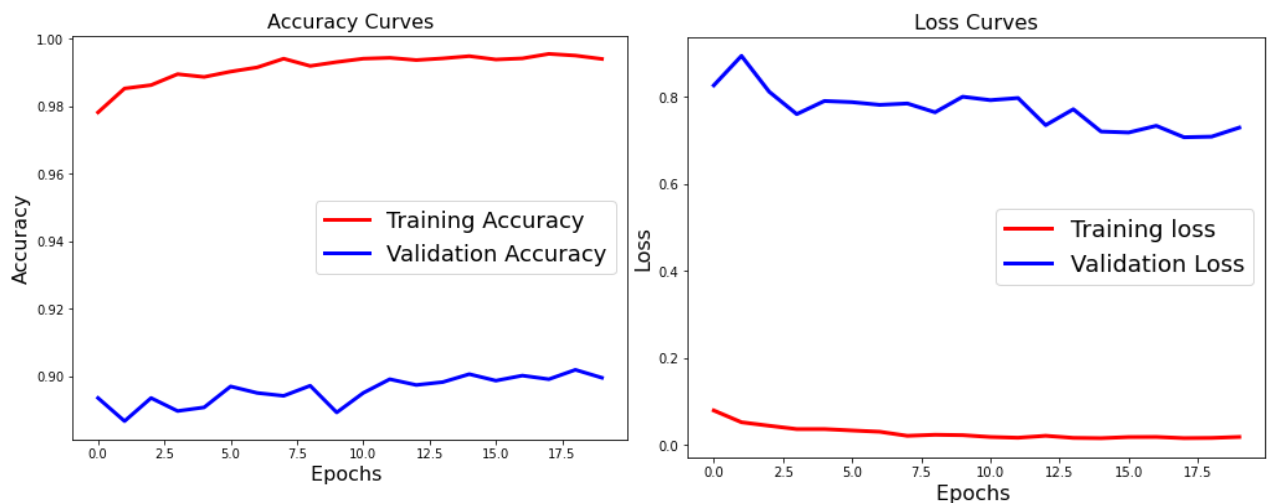


OPTIMIZER='SGD'

NN Prediction 0.8985228002569043

Classification report

	CLASS	precision	recall	f1-score	support
	0	0.96	0.97	0.97	157
	1	0.94	0.92	0.93	163
	2	0.89	0.91	0.90	175
	3	0.93	0.88	0.90	186
	4	0.92	0.92	0.92	169
	5	0.88	0.88	0.88	188
	6	0.96	0.85	0.90	157
	7	0.85	0.94	0.90	174
	8	0.93	0.83	0.88	166
	9	0.86	0.91	0.89	162
	10	0.86	0.86	0.86	168
	11	0.84	0.93	0.89	170
	12	0.91	0.93	0.92	162
	13	0.89	0.87	0.88	155
	14	0.93	0.86	0.89	189
	15	0.88	0.91	0.89	184
	16	0.95	0.84	0.89	166
	17	0.82	0.93	0.87	160
	18	0.96	0.92	0.94	164
	19	0.80	0.84	0.82	160
	20	0.94	0.79	0.86	182
	21	0.92	0.88	0.90	158
	22	0.95	0.96	0.96	160
	23	0.95	0.95	0.95	171
	24	0.82	0.90	0.86	157
	25	0.94	0.95	0.94	171
	26	0.93	0.93	0.93	152
	27	0.92	0.91	0.91	145
	micro avg	0.90	0.90	0.90	4671
	macro avg	0.90	0.90	0.90	4671
	weighted avg	0.90	0.90	0.90	4671
	samples avg	0.90	0.90	0.90	4671



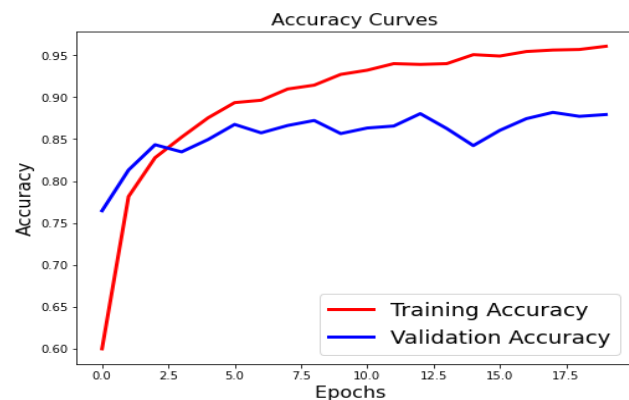
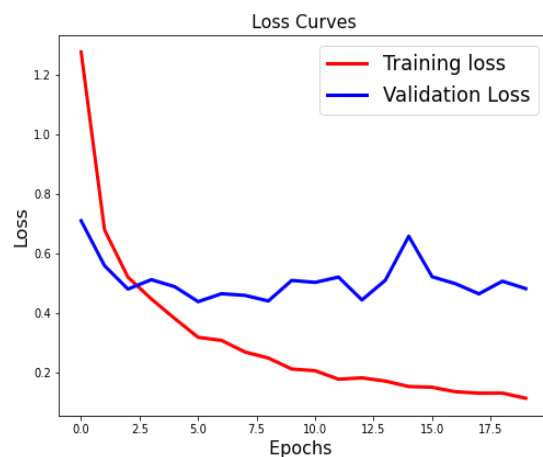
OPTIMIZER='ADAM'

NN Prediction 0.8724041961036181

Classification report

CLASS	precision	recall	f1-score	support
0	0.98	0.97	0.97	157
1	0.89	0.91	0.90	163
2	0.84	0.93	0.89	175
3	0.93	0.82	0.87	186
4	0.93	0.85	0.89	169
5	0.83	0.89	0.86	188
6	0.93	0.83	0.88	157
7	0.88	0.90	0.89	174
8	0.89	0.81	0.85	166
9	0.87	0.85	0.86	162
10	0.83	0.82	0.82	168
11	0.89	0.87	0.88	170
12	0.95	0.85	0.90	162
13	0.87	0.88	0.87	155
14	0.91	0.82	0.86	189
15	0.87	0.90	0.89	184
16	0.92	0.89	0.90	166
17	0.81	0.93	0.86	160
18	0.92	0.93	0.92	164
19	0.77	0.79	0.78	160
20	0.91	0.76	0.83	182
21	0.88	0.90	0.89	158
22	0.96	0.94	0.95	160
23	0.99	0.88	0.93	171
24	0.85	0.84	0.84	157

	25	0.97	0.87	0.92	171
	26	0.92	0.90	0.91	152
	27	0.83	0.92	0.87	145
micro avg		0.89	0.87	0.88	4671
macro avg		0.89	0.87	0.88	4671
weighted avg		0.89	0.87	0.88	4671
samples avg		0.87	0.87	0.87	4671



Hybrid Models

Hybrid Models are those models which comprise of two or more models which are integration of ML models and other soft computing , deep learning or optimization techniques .

In this proposal we have used CNN as a base feature extractor and the feature extracted is then fed into a Machine Learning model and see how the model performs . The architecture for the feature extractor is the same as the CNN model and various Machine Learning models are used.

1. CNN+SVM

KERNEL : RBF

CLASSIFICATION REPORT FOR ARABIC MNIST

CLASS	precision	recall	f1-score	support
0	0.93	0.97	0.95	106
1	0.98	0.98	0.98	96
2	0.95	0.98	0.97	104
3	0.99	0.98	0.98	88
4	1.00	0.99	1.00	109
5	0.93	0.93	0.93	108
6	0.99	0.99	0.99	101
7	0.99	0.97	0.98	97
8	0.99	0.97	0.98	86
9	0.97	0.96	0.97	105
accuracy			0.97	1000
macro avg	0.97	0.97	0.97	1000
weighted avg	0.97	0.97	0.97	1000

Accuracy: 0.971

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

CLASS	precision	recall	f1-score	support
0	0.88	0.92	0.90	38
1	0.82	0.94	0.87	33
2	0.86	0.91	0.89	35
3	0.84	0.84	0.84	38
4	0.97	0.94	0.95	33
5	0.89	0.93	0.91	42
6	0.89	0.89	0.89	35
7	0.85	0.87	0.86	39
8	0.78	0.83	0.81	30
9	0.86	0.94	0.90	32
10	0.90	0.85	0.88	33
11	0.79	0.86	0.82	35
12	0.93	0.88	0.90	32
13	0.84	0.87	0.86	31
14	0.83	0.85	0.84	34

15	0.91	0.87	0.89	47	
16	0.91	0.82	0.86	38	
17	0.77	0.89	0.83	38	
18	0.91	0.82	0.86	39	
19	0.77	0.75	0.76	36	
20	0.93	0.86	0.89	43	
21	0.88	0.82	0.85	34	
22	0.97	1.00	0.98	30	
23	0.98	0.91	0.94	44	
24	0.88	0.75	0.81	28	
25	0.89	0.89	0.89	35	
26	0.83	0.87	0.85	39	
27	0.92	0.83	0.87	29	
accuracy				0.87	1000
macro avg		0.87	0.87	0.87	1000
weighted avg		0.87	0.87	0.87	1000

Accuracy: 0.872

2. CNN+ RANDOM FOREST

CLASSIFICATION REPORT FOR ARABIC MNIST

random forest 0.939

Classification report

CLASS	precision	recall	f1-score	support	
0	0.98	0.92	0.95	106	
1	0.94	0.97	0.95	96	
2	0.92	0.88	0.90	104	
3	0.91	0.98	0.94	88	
4	0.92	0.94	0.93	109	
5	0.89	0.94	0.91	108	
6	0.94	0.97	0.96	101	
7	1.00	0.97	0.98	97	
8	0.99	0.90	0.94	86	
9	0.92	0.92	0.92	105	
accuracy				0.94	1000
macro avg		0.94	0.94	0.94	1000

weighted avg	0.94	0.94	0.94	1000
--------------	------	------	------	------

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

random forest 0.804

Classification report

CLASS	precision	recall	f1-score	support
0	0.92	0.92	0.92	38
1	0.65	0.91	0.76	33
2	0.77	0.77	0.77	35
3	0.88	0.74	0.80	38
4	0.97	0.88	0.92	33
5	0.88	0.88	0.88	42
6	0.77	0.77	0.77	35
7	0.85	0.87	0.86	39
8	0.76	0.83	0.79	30
9	0.68	0.84	0.75	32
10	0.77	0.82	0.79	33
11	0.75	0.77	0.76	35
12	0.92	0.75	0.83	32
13	0.81	0.81	0.81	31
14	0.80	0.71	0.75	34
15	0.81	0.74	0.78	47
16	0.76	0.68	0.72	38
17	0.65	0.84	0.74	38
18	0.90	0.72	0.80	39
19	0.80	0.67	0.73	36
20	0.90	0.84	0.87	43
21	0.81	0.85	0.83	34
22	0.93	0.93	0.93	30
23	0.93	0.89	0.91	44
24	0.90	0.64	0.75	28
25	0.76	0.80	0.78	35
26	0.66	0.79	0.72	39
27	0.75	0.83	0.79	29
28	0.00	0.00	0.00	0
micro avg	0.80	0.80	0.80	1000
macro avg	0.78	0.78	0.78	1000
weighted avg	0.81	0.80	0.80	1000

CNN+ ADABOOST

CLASSIFICATION REPORT FOR ARABIC MNIST

Ada boost 0.547

Classification report

	CLASS	precision	recall	f1-score	support
	0	0.98	0.80	0.88	106
	1	0.83	0.79	0.81	96
	2	0.45	0.63	0.52	104
	3	0.30	0.64	0.41	88
	4	0.48	0.37	0.42	109
	5	0.48	0.36	0.41	108
	6	0.71	0.75	0.73	101
	7	0.56	0.49	0.52	97
	8	0.48	0.57	0.52	86
	9	0.46	0.11	0.18	105
	accuracy			0.55	1000
	macro avg	0.57	0.55	0.54	1000
	weighted avg	0.58	0.55	0.54	1000

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

Ada boost 0.169

Classification report

	CLASS	precision	recall	f1-score	support
	0	0.74	0.37	0.49	38
	1	0.10	0.12	0.11	33
	2	0.00	0.00	0.00	35
	3	0.26	0.16	0.20	38
	4	0.25	0.06	0.10	33
	5	0.23	0.26	0.24	42
	6	0.16	0.63	0.26	35
	7	0.00	0.00	0.00	39
	8	0.06	0.47	0.10	30
	9	0.22	0.56	0.32	32
	10	0.24	0.58	0.34	33
	11	0.00	0.00	0.00	35
	12	0.00	0.00	0.00	32

13	0.10	0.29	0.15	31
14	0.21	0.18	0.19	34
15	0.00	0.00	0.00	47
16	0.21	0.16	0.18	38
17	0.00	0.00	0.00	38
18	0.21	0.18	0.19	39
19	0.00	0.00	0.00	36
20	0.00	0.00	0.00	43
21	0.13	0.06	0.08	34
22	0.43	0.33	0.38	30
23	0.26	0.20	0.23	44
24	0.00	0.00	0.00	28
25	0.00	0.00	0.00	35
26	0.21	0.23	0.22	39
27	0.14	0.03	0.06	29
28	0.00	0.00	0.00	0
micro avg	0.17	0.17	0.17	1000
macro avg	0.14	0.17	0.13	1000
weighted avg	0.15	0.17	0.14	1000

CNN+XGBOOST

CLASSIFICATION REPORT FOR ARABIC MNIST

XG boost 0.932

Classification report

CLASS	precision	recall	f1-score	support
0	0.98	0.95	0.97	106
1	0.98	0.96	0.97	96
2	0.88	0.88	0.88	104
3	0.90	0.94	0.92	88
4	0.91	0.94	0.92	109
5	0.92	0.88	0.90	108
6	0.99	0.96	0.97	101
7	0.99	0.97	0.98	97
8	0.86	0.93	0.89	86
9	0.91	0.92	0.92	105
accuracy			0.93	1000
macro avg	0.93	0.93	0.93	1000
weighted avg	0.93	0.93	0.93	1000

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

XG boost 0.799

Classification report

	CLASS	precision	recall	f1-score	support
	0	0.95	0.92	0.93	38
	1	0.70	0.91	0.79	33
	2	0.80	0.80	0.80	35
	3	0.83	0.79	0.81	38
	4	0.97	0.91	0.94	33
	5	0.90	0.86	0.88	42
	6	0.84	0.77	0.81	35
	7	0.84	0.82	0.83	39
	8	0.67	0.73	0.70	30
	9	0.77	0.84	0.81	32
	10	0.68	0.82	0.74	33
	11	0.72	0.83	0.77	35
	12	0.96	0.72	0.82	32
	13	0.75	0.68	0.71	31
	14	0.83	0.71	0.76	34
	15	0.81	0.64	0.71	47
	16	0.81	0.79	0.80	38
	17	0.67	0.92	0.78	38
	18	0.86	0.77	0.81	39
	19	0.56	0.67	0.61	36
	20	0.84	0.74	0.79	43
	21	0.85	0.82	0.84	34
	22	0.93	0.90	0.92	30
	23	0.90	0.84	0.87	44
	24	0.81	0.75	0.78	28
	25	0.74	0.83	0.78	35
	26	0.72	0.79	0.76	39
	27	0.92	0.83	0.87	29
	28	0.00	0.00	0.00	0
	micro avg	0.80	0.80	0.80	1000
	macro avg	0.78	0.77	0.77	1000
	weighted avg	0.81	0.80	0.80	1000

CNN+ CATBOOST

CLASSIFICATION REPORT FOR ARABIC MNIST

CatBoost 0.926

Classification report			precision	recall	f1-score
support					
0	0.97	0.92	0.94	106	
1	0.95	0.97	0.96	96	
2	0.91	0.90	0.91	104	
3	0.85	0.95	0.90	88	
4	0.90	0.93	0.91	109	
5	0.90	0.88	0.89	108	
6	0.96	0.98	0.97	101	
7	0.99	0.92	0.95	97	
8	0.90	0.91	0.90	86	
9	0.93	0.91	0.92	105	
accuracy			0.93	1000	
macro avg	0.93	0.93	0.93	1000	
weighted avg	0.93	0.93	0.93	1000	

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

bestTest = 0.9272568773

bestIteration = 99

CatBoost 0.795

Classification report					
CLASS	precision	recall	f1-score	support	
0	0.89	0.89	0.89	38	
1	0.62	0.94	0.75	33	
2	0.70	0.74	0.72	35	
3	0.84	0.71	0.77	38	
4	0.86	0.91	0.88	33	
5	0.85	0.79	0.81	42	
6	0.84	0.77	0.81	35	
7	0.82	0.79	0.81	39	
8	0.74	0.77	0.75	30	
9	0.77	0.94	0.85	32	
10	0.71	0.82	0.76	33	
11	0.82	0.80	0.81	35	

12	0.90	0.88	0.89	32
13	0.93	0.81	0.86	31
14	0.73	0.79	0.76	34
15	0.78	0.77	0.77	47
16	0.83	0.66	0.74	38
17	0.68	0.84	0.75	38
18	0.92	0.85	0.88	39
19	0.68	0.53	0.59	36
20	0.82	0.74	0.78	43
21	0.80	0.82	0.81	34
22	0.93	0.87	0.90	30
23	0.95	0.89	0.92	44
24	0.83	0.71	0.77	28
25	0.72	0.80	0.76	35
26	0.62	0.67	0.64	39
27	0.89	0.83	0.86	29
28	0.00	0.00	0.00	0
micro avg	0.80	0.80	0.80	1000
macro avg	0.78	0.77	0.77	1000
weighted avg	0.80	0.80	0.80	1000

CNN+ LOGISTIC REGRESSION

CLASSIFICATION REPORT FOR ARABIC MNIST

Logistic Regression 0.938

Classification report

	precision	recall	f1-score	support
0	0.95	0.94	0.95	106
1	0.99	0.99	0.99	96
2	0.89	0.89	0.89	104
3	0.89	0.97	0.92	88
4	0.89	0.95	0.92	109
5	0.94	0.88	0.91	108
6	0.97	0.98	0.98	101
7	1.00	0.94	0.97	97
8	0.98	0.93	0.95	86
9	0.91	0.91	0.91	105
accuracy			0.94	1000

macro avg	0.94	0.94	0.94	1000
weighted avg	0.94	0.94	0.94	1000

CLASSIFICATION REPORT FOR ARABIC CHARACTER DATASET

Logistic Regression 0.856

Classification report

	precision	recall	f1-score	support
0	0.88	0.92	0.90	38
1	0.88	0.91	0.90	33
2	0.80	0.80	0.80	35
3	0.78	0.84	0.81	38
4	0.93	0.85	0.89	33
5	0.84	0.88	0.86	42
6	0.86	0.86	0.86	35
7	0.85	0.90	0.88	39
8	0.75	0.80	0.77	30
9	0.82	0.88	0.85	32
10	0.79	0.79	0.79	33
11	0.83	0.86	0.85	35
12	0.91	0.91	0.91	32
13	0.77	0.77	0.77	31
14	0.88	0.85	0.87	34
15	0.91	0.85	0.88	47
16	0.91	0.82	0.86	38
17	0.79	0.89	0.84	38
18	0.91	0.79	0.85	39
19	0.75	0.75	0.75	36
20	0.86	0.86	0.86	43
21	0.93	0.82	0.87	34
22	0.88	1.00	0.94	30
23	0.98	0.91	0.94	44
24	0.96	0.79	0.86	28
25	0.83	0.86	0.85	35
26	0.85	0.87	0.86	39
27	0.87	0.93	0.90	29
28	0.00	0.00	0.00	0
micro avg	0.86	0.86	0.86	1000
macro avg	0.83	0.83	0.83	1000
weighted avg	0.86	0.86	0.86	1000

Conclusion

FOR ARABIC MNIST

Model	Accuracy Score
Random Forest	0.98714
CatBoost	0.95476
SVM (linear)	0.99628
SVM(RBF)	0.99628
Adaboost	0.696
XGBOOST	1.0000
LOGISTIC REGRESSION	0.99957
Feedforward Neural Network	0.98720

CNN	0.990625
Transfer Learning	0.996743
CNN+SVM	0.9710
CNN+RF	0.9390
CNN+ADABOOST	0.5470
CNN+ XGBOOST	0.9320
CNN+ LR	0.9388

FOR ARABIC CHARACTER DATASET

Model	Accuracy Score
Random Forest	0.6335
CatBoost	0.4642
SVM (linear)	0.4233
SVM(RBF)	0.6528
XGBOOST	0.4854
LOGISTIC REGRESSION	0.6528
Feedforward Neural Network	0.6101
CNN	0.8991
CNN+SVM	0.8772

CNN+RF	0.8004
CNN+ADABOOST	0.1690
CNN+ XGBOOST	0.7990
CNN+ LR	0.8560

Thus we conclude that for ARABIC MNIST dataset , Logistic Regression Model and XGBOOST model outperforms all the models in the research field in this area, the previous record of 99.71% is outperformed by our Machine Learning approach. Hybrid models like CNN+SVM have reached near accuracy however, hybrid models are more robust than standalone cnn models. Higher Recall rate is observed in hybrid models. Out of all models which have performed worse are Adaboost and CNN+ADABOOST models.

For ARABIC CHARACTER dataset , Standalone CNN models has outperformed all the models and also all the models in this research. The reason for 90% accuracy and not hitting higher accuracy like MNIST values is because of the large class size and interclass difference being very less and higher variance . This thus makes the model bottleneck while doing boundary decisions at the feature map. The worst performing models were Feedforward neural networks with sgd optimizer. The model fails to converge and remains at plateau. Machine learning model didn't perform well and however in the Hybrid models, we see the model performing well. XGBoost single handedly didn't perform well but with CNN+XGBOOST , the accuracy jumps from 0.4854 to 0.7990. The worst machine learning model being the SVM with linear kernel.

The XGBOOST outperforms the previous models in paper

<https://www.sciencedirect.com/science/article/pii/S1877050920307754>,

<https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8863484>

References

1. [CNN for Handwritten Arabic Digits Recognition Based on LeNet-5](#)
2. [Arabic handwritten digit recognition](#)
3. [Handwritten Arabic numerals recognition using convolutional neural network](#)
4. [Comparing Arabic and Latin Handwritten Digits Recognition Problems](#)
5. [Arabic Handwritten Characters Classification Using Learning Vector Quantization Algorithm](#)
6. [Arabic Handwritten Characters Recognition using Convolutional Neural Network](#)
7. [On Arabic Character Recognition Employing Hybrid Neural Network](#)
8. [A New Design Based-SVM of the CNN Classifier Architecture with Dropout for Offline Arabic Handwritten Recognition](#)
- 9.

