42028: Deep Learning and Convolutional Neural Network

Student Name: Nazmul Kaonine

Student ID: 13300912

1. Introduction:T

This report focuses on the MNIST database of handwritten digits and the development of three different classifiers in order to identify these numbers. To achieve this, traditional machine learning methods such as K-Nearest Neighbour (KNN), Support Vector Machine (SVM) and Multi-layer Perceptron Neural Network (MLP) have been used. Feature extraction techniques such as Histogram of Oriented Gradient (HOG), Linear Binary Pattern (LBP), Principle Component Analysis (PCA) and simple raw inputs have been applied to each of these classifiers. To observe the combination with the highest accuracy, different parameters have been tweaked and the outlining reasons behind high and low accuracy scores have been investigated.

Combination Descriptions-

HOG: First used in identifying pedestrians, HOG is an important feature extraction method that computes histograms of gradients based on gradient magnitude and direction.

LBP: This texture operator efficiently labels each pixel of an image by thresholding its neighbors.

PCA: Reduces the dimensions of the dataset while keeping variation inside the data as much as possible. It is used in image compression as it filters noisy datasets.

Classifiers-

SVM: A set of supervised learning methods used namely for outliers detection, regression and classification. In a n-dimensional space where n is the number of features, the data points are plotted and a hyperplane separates the two classes.

KNN: The simplest image classifier that uses the distance between feature vectors. It identifies unknown data by through commonality among the closest learned samples.

MLP: A supervised algorithm that learns from a dataset. There can be multiple non-linear hidden layers between the input-output computation of this algorithm. The function that it learns-

F(.): R^m->R^o

m : number of input dimensions,

o : number of output dimensions

2. Dataset:

This report focuses on identifying numbers from the MNIST Handwritten Digits Dataset. This dataset was made from the National Institute of Standards & Technology's Special Database 3 and Special Database 1 which consists of handwritten numbers in binary image format. The data set used has been downloaded from the link http://yann.lecun.com/exdb/mnist/ The data is not in any standard image format, therefore the train and test files have been formatted first.

The deconstruction of the dataset is as follows:

Total images: 70,000 (60,000 training+10,000 testing)

Labels: [0,1,2,3,4,5,6,7,8,9]

Images size: 28x28 pixels (grayscale)

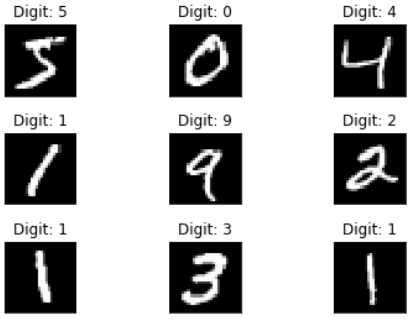

Fig:1 Sample images of each class

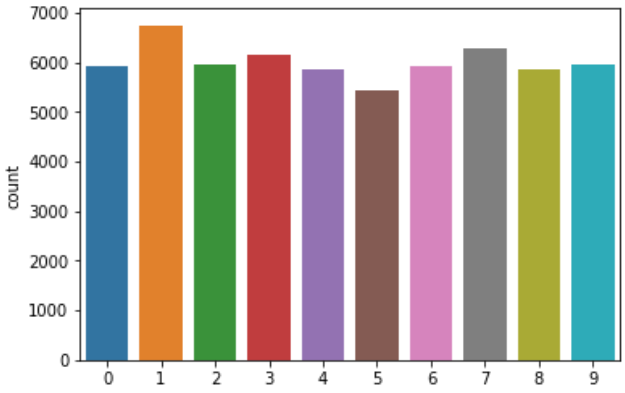

Fig:2 Count of each label

The dataset was in .gz format so gzip has been used to open each of the four files and loaded under train and test labels. To visualize the dataset, the shapes had to be changed e.g. (60,000,784) training data was reshaped to (60,000, 28,28) to achieve image pixel readable format.

3. Proposed CNN architecture for Image Classification:

a. Baseline architecture used.
b. Customized architecture
c. Assumptions/intuitions
d. Model summary

4. CNN Architecture for Object Detection/Localization:

a. Faster-RCNN.
[ii. Better accuracy add or delete layers and why, try ssd 512 shape]
b. SSD (Single Shot Detector)
c. Assumptions/intuitions
d. Model summary

5. Experimental results and discussion:

a. Experimental settings:

It is evident from this report that the accuracy is dependant on the parameters used for each classifier. All accuracies are test accuracies and the best settings are shown. Relevant information on the classifier settings are listed below. To even out the competition, test sample = 0.1 was used.

i. Image classification
After using a loop on the classifier from k=1 to 15 on a test sample of smaller size for quicker classification, it was observed:

algorithm='auto', **leaf_size**=30, **metric**='minkowski', **metric_params**=None, **n_jobs**=None, **n_neighbors**=k, p=2, **weights**='uniform'

K values	**Accuracy**	**Feature**
1……15	**0.97**	**Raw Input (k =5)**
1...15	**0.8792**	Hog
1...15	**0.40...0.60**	LBP

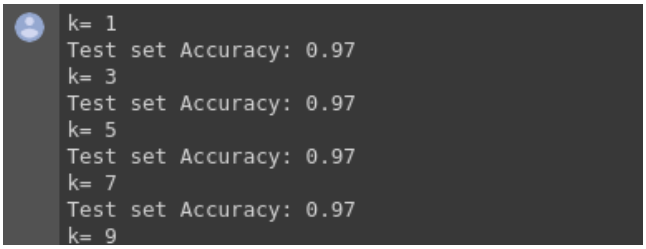

Fig 3: KNN accuracies

HOG: Best among the descriptors, accuracy received = 0.9095
LBP: Achieved lowest accuracy of 0.4369
PCA: Received a decent accuracy of 0.833

Note: It is important to try multiple K-values to find the optimal number of votes for each data point classification. A loop was used in this experiment between (1,30)

ii. Object Detection:

Kernel	**C**	**Gamma**	**Degree**	**Max Accuracies**
linear				0.97 (Raw Input)
rbf	100..500	0.01...0.05		0.9792 (C=100)(G=0.01) (Raw Input)
poly	100..500	0.01...0.05	(2….6)	**0.9833 (degree=4, gamma=0.01, C=100, Raw Input)**

HOG: Upon Using the raw input settings a much lesser accuracy was received.
LBP: Accuracy declined by 9% from raw input accuracy.
PCA:

Note:
The classification kernel focused on is linear due to the visible edge detection in the numerals. But upon raising dimensions i.e. rbf or Gaussian model, it resulted a better accuracy. Poly accuracies were not increasing.

The value of C (number of data samples per each class and Gamma (influence of each sample) was tweaked and the highest accuracy (C=200, gamma=0.01) is noted.
When using **kernel**= 'poly', **degree**=4 was used to determine the hyperplane.

ANN:

Input Shape: none, **Model:** Sequential, **Layer:** Dense**Loss :** 'sparse_categorical_crossentropy'

Neuron #	**Activation Function Relu**	**Activation Function Softmax**	**Epochs**	**Optimizer**	**Max Accuracy**
128, 10	128	10	10	adam	0.97
128,10	128	10	50.	adam	**0.9794**
128,20	128	20	50..500	adam	**0**.9698
256,10	256	10	50..500	adam	0.9664
256,20	128	10	50...500	Adamax	0.97
128, 10	128	10	50..500	SGD	0.97
128,10	128	10	50	Adadelta	0.93
128,10	128	10	50	Nadam	0.97

LBP: Max Accuracy 0.53 Upon same settings
Hog: Max Accuracy (Adam: 0.91 SGD: 0.79)
PCA:

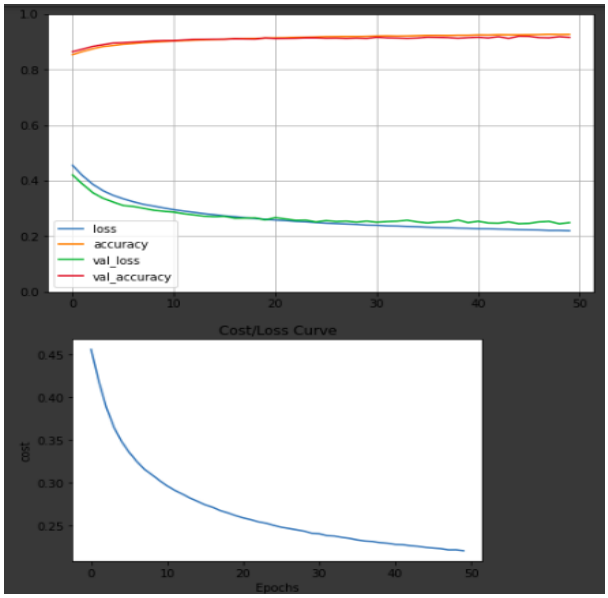

Fig: 4
Cost/ Loss & Epoch Curve for HOG (ANN)

```
313/313 [==============================] - 0s 1ms/step - loss: 0.1504 - accuracy: 0.9794
[0.15040640532970428, 0.9793999791145325]
```

Fig:5 Max ANN accuracy score

Note: Whereas varying the optimizer does change the accuracy significantly, varying the number of nodes has lesser impact. Also, keeping the input shape to none provided the flexibility to change features.

b. **Experimental results:**

I. Image classification:

1. Performance on baseline/ standard architecture

2. Performance on customized architecture

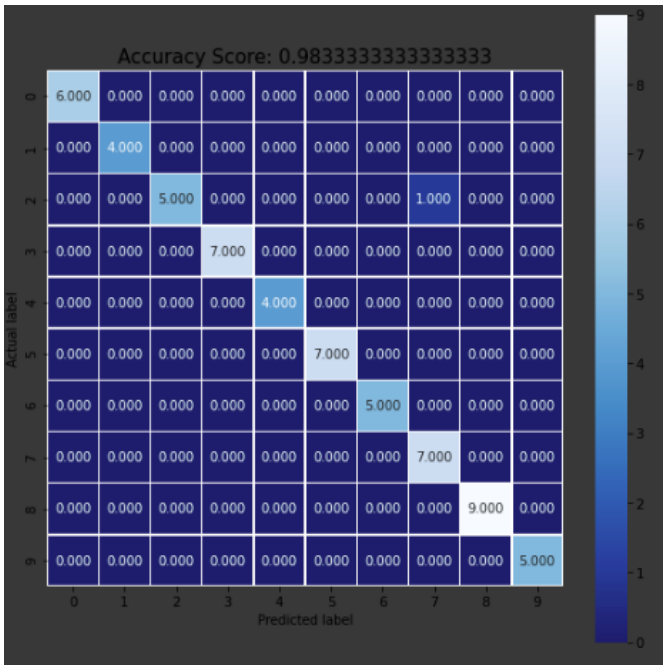

Fig 6: Confusion matrix of the highest accuracy 98.33%
SVM (kernel='poly', degree=4, C=100, gamma=0.01)

The confusion matrix shows the digit '2' as the most correctly classified (1128 times). The most wrongly classified digits are "4" as "9" and the number "7" as "2". Nevertheless, these digits do look alike in the images.

Fig: 7 Some correctly classified result images

ii. Object detection

1.Performance on Faster-RCNN

2. Performance on SSD or customized architecture

Classifier/Feature	HOG	LBP	Raw Input
KNN	0.8676	0.4369	0.97
SVM	0.9095	0.5219	**0.9833**
ANN	0.9166	0.53	0.9794

Iii. Discussion:

Reasons behind the classifier accuracies are outlined:

SVM

Firstly, SVM is well known for doing binary classification. But since the problem has 10 separable classes, other kernel function was tested to take the classification from linear to higher dimensional space, thus resulting in an effective classifier.

With the radial basis function, using the power of slack [C] that allows certain outliers to be in the wrong side of the boundary, an attempt to reduce overfitting was made as SVM hyperplane is only affected by support vectors. Additionally, a suitable gamma value to increase the radius of influence on the training dataset gave a near 98% accuracy. But when these values in addition to kernel poly with degree=4 (increase in degree of the polynomial function), it gives us the highest accuracy among all of the classifiers.

ANN

Neural networks are specially prone to overfitting. It generalizes the training data so much that it is less effective in the unknown. In simple terms, each hidden layer transform all the features into one single value thus reducing the context of the data. Although, this can lead to the little difference in accuracy, the accuracy produced is still 0.9794.

One of the reason is an increase in the number of nodes resulted in more tunable parameters. Thus with a moderate number of layers, a good number of neurons, the epoch value was increased to 50. Rather than using SGD that has a simple learning rate, optimizer 'adam' was used to form an weight-focused iterative learning procedure resulting in the second-best accuracy.

KNN

This can be a sensitive classifier as it may underfit the data when the k-values are too large by making indistinct decision boundaries. When smaller k-values are used, overfitting occurs, resulting in more noise. The k-values will always have some degree of noise thus resulting in the third best accuracy.

LBP

The reason why LBP consistently gave a low accuracy is because it is a primary texture-based classifier. It looks at points surrounding the central point and determines whether the surrounding points are greater or less than the central point. As the mnist digits have distinct colors for the numbers and the background, many digit data points will be merged into the background and many background points may get merged inside the digits.

HOG

The reason why HOG gave a better accuracy is because it divides the images into small square cells. The histograms are derived from edge directions or intensity gradients. In this experiment, the features are extracted from all locations from the digit image resulting in a good accuracy value.

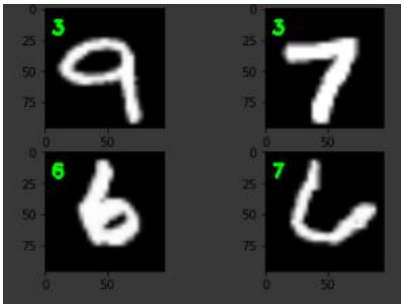

Fig 8: A few wrong classifications from LBP (KNN)

PCA: This technique helps the classifiers decide the angle to be used to look at the 784-dimensional cube. It captures the most variance as possible. Simultaneously, it reduces the dimensions to find each of the components resulting in a decent accuracy.

6. Conclusion

The observation from the experiments is that SVM when tweaked to an optimal level is outstanding for classifying digits and so are neural networks with an effective optimizer. KNNs with a straighforwad tuning parameter performed well but not the best. It might be a good decision to stay away from LBP while performing experiments on the MNIST data. Nevertheless, HOG and PCA are decent scorers. Although SVM with a higher dimensional kernel gives the best accuracy, it does take more time than the others. In the future, it will be worthwhile to see how a CNN and virtual SVM work on this dataset.