Evaluation of machine learning classification schemes for MNIST and ORL datasets

Jayendra Ellamathy, *Masters in Computer Engineering, Aarhus Universitet*

***Abstract*—In this paper we evaluate the performance of 5 classification algorithms on the MNIST and the ORL dataset. These 5 algorithms are - Nearest Centroid Classifier, Nearest Sub-class Centroid Classifier, Nearest Neighbor Classifier, Perceptron trained with Backpropagation, Perceptron trained with MSE(least squares solution).**

***Index Terms*—ORL, MNIST, nearest centroid classifier, nearest neighbor classifier, nearest sub-class classifier, perceptron with backpropagation, perceptron with MSE.**

I. Introduction

In machine learning, there are two types of learning - supervised and unsupervised. Classification falls under the category of supervised learning. In supervised learning, pre-labeled samples are provided to the ML model, and the model in turn learns from these samples called the training samples. The ML model tunes its parameters from the training samples. The way the parameters are tuned depends on the ML model used. There are several classification models, among which 5 popular models are evaluated here. They are - Nearest Centroid Classifier, Nearest Sub-class Centroid Classifier, Nearest Neighbor Classifier, Perceptron trained with Backpropagation, Perceptron trained with MSE(least squares solution). From the trained model we evaluate the performance by providing it test samples. Here, we know the labels of the test samples and we predict the samples using our trained model and compare the two to get a sense of accuracy. Various classification models have their advantages and disadvantages and are better suited for certain kinds of datasets over others. Here, two image datasets - the ORL(faces) and MNIST(hand-written digits from 0 - 9) are used to compare differnces.

II. Description of classification schemes used

A. Nearest Class Centroid(NCC) Classifier

In this classification model, an observation is labeled to the class of training samples whose mean (centroid) is closest to the observation. The algorithm for the Nearest Centroid Classifier is as follows:

- Training procedure: given labeled training samples $\{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\}$ with class labels $y_i \in \mathbf{Y}$, compute the per-class centroids: $\vec{\mu}_l = \frac{1}{|C_l|}\sum_{i \in C_l} \vec{x}_i$ where C_l is the set of indices of samples belonging to class $l \in \mathbf{Y}$.
- Prediction function: the class assigned to an observation $\vec{x}$ is $\hat{y} = \arg\min_{l \in \mathbf{Y}} \|\vec{\mu}_l - \vec{x}\|$.

B. Nearest Subclass Centroid(NSC) Classifier

The Nearest Subclass Classifier (NSC), which is a classification algorithm that unifies the flexibility of the nearest neighbor classifier with the robustness of the nearest mean classifier.
Given a set of N samples, each represented by a $\vec{x}_i \in R^D$, and the corresponding labels l_i. Here, each class is divided into subclasses. NSC assumes each subclass m of class c_k follows a normal density and is represented by the mean subclass vector μ_{km}:

$$\mu_{km} = \frac{1}{N_{km}} \sum_{i, l_i = k, q_i = m} x_i$$

Here we use the label q_i in order to denote the subclass label of $\vec{x}_i$ and N_{km} to denote the number of samples forming subclass m of class c_k. Given the subclass mean vectors, x_* is classified to the class c_k corresponding to the smallest distance:

$$d(x_*, \mu_{km}) = ||x_* - \mu_{km}||_2^2$$

C. Nearest Neighbor(NN) Classifier

The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples can be a user-defined constant (k-nearest neighbor learning), or vary based on the local density of points (radius-based neighbor learning). The distance can, in general, be any metric measure: standard Euclidean distance is the most common choice.

D. Perceptron trained with backpropagation

Backpropogation is a widely used algorithm for training feedforward neural networks. The backpropagation algorithm works by computing the gradient of the loss function with respect to each weight by the chain rule, computing the gradient one layer at a time, iterating backward from the last layer to avoid redundant calculations of intermediate terms in the chain rule.

E. Perceptron trained with MSE(least squares solution)

Mean square error (MSE) is calculated by taking an average of the sum of the squared difference between predicted and actual values. MSE gives the average magnitude of error irrespective of their direction.
MSE value gets highly modified when there is a small change indifference of prediction and actual value. Due to squaring, it penalizes heavily to the model as per the difference between predicted and actual value. It is easier to calculate gradients with MSE.

III. Databases description

A. The MNIST dataset

This dataset comprises of images of hand-written digits. The dataset is split into 60,000 samples of training data and 10,000 samples of testing data. The images are represented by (28x28) matrix with each element taking values between 0 - 1, for representing varying levels of gray. It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting. The MNIST database can be obtained from here [1].

Fig. 1: Sample images from the MNIST dataset

B. The ORL dataset

The ORL Database of Faces contains 400 images from 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). the ORL database can be accessed from here [2].

Fig. 2: Sample images from the ORL database

IV. Experimental protocol

All the 5 ML models were implemented with the sklearn[3] library from python[4].

A. Pre-processing data

An image in the MNIST dataset is a 28x28 matrix which is flattened into a 784D vector. Similarly, in the ORL dataset an image is flattened into a 1200D vector. PCA is then applied to the dataset to reduce the dimensions to 2. Both the original dataset and the PCA reduced dataset are applied for the various ML algorithms.

B. ML algorithm implementation from sci-kit learn

1) NCC: NearestCentroid() from the sklearn library is used.
2) NSC: For the NSC, first all the samples are segregated by their labels and then K-means(KMeans(n_clusters=K)) is applied on each sample subset with the number of subclasses K = 2, 3, 5. From the centroids obtained from each sub-class, we predict using the nearest neighbor of the centroids of the sub-class using KNeighborsClassifier(n_neighbors=1).
3) NN: KNeighborsClassifier(n_neighbors=K) from the sklearn library is used. The method is tested with neighbor numbers of 1, 2, and 3.
4) Perceptron trained with backpropagation: SGDClassifier(loss=”hinge”) from the sklearn library is used.
5) Perceptron trained with MSE: SGDClassifier(loss=”squared_error”) from the sklearn library is used.

C. 2D visualisation of datasets

Since it is not possible to view higher dimensional data such as the MNIST and ORL, we need to reduce the dimensions. This is done here using two different methods - PCA and t-SNE, and then plotted on matplotlib[5]. The visualisation is shown in Fig 3 and 4.

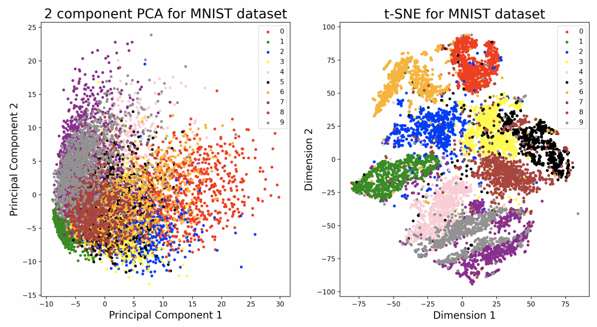

Fig. 3: 2D visualisation of the MNIST dataset

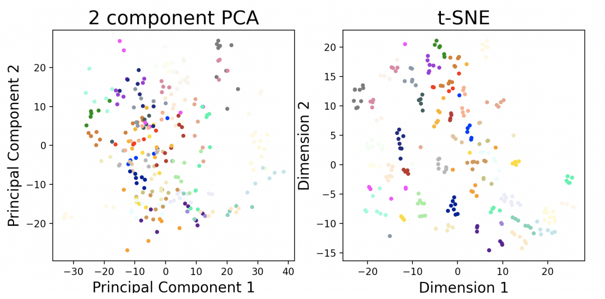

Fig. 4: 2D visualisation of the ORL dataset

V. Experimental results and discussion

A. Nearest Centroid Classifier

ORL:

- Accuracy: 88.33%
- Accuracy(PCA reduced): 35.83%

MNIST:

- Accuracy: 82.03%
- Accuracy(PCA reduced): 34.83%

B. Nearest Subclass Centroid Classifier

ORL:

- Accuracy for sub classes 2, 3, 5: 94, 95, 95 %
- Accuracy for sub classes 2, 3, 5(PCA reduced): under 10%

MNIST:

- Accuracy for sub classes 2, 3, 5: 86.1, 88.15, 90.32 %
- Accuracy for sub classes 2, 3, 5(PCA reduced): 34.69, 34.45, 32.7 %

C. Nearest Neighbor Classifier

ORL:

- Accuracy with K = 1, 2, 3: 95, 83.33, 90 %
- Accuracy with K = 1, 2, 3(PCA reduced): 36.66, 33.33, 30.83 %

MNIST:

- Accuracy with K = 1, 2, 3: 96.91, 96.27, 97.05 %
- Accuracy with K = 1, 2, 3(PCA reduced): 29.21, 30.84, 31.55 %

D. Perceptron trained with backpropagation

ORL:

- Accuracy with iterations 10, 100 = 82.5, 85.83 %
- Accuracy with iteration 10, 100(PCA reduced) = Under 10%

MNIST

- Accuracy with iterations 10, 100 = 91.34, 91.13 %
- Accuracy with iteration 10, 100(PCA reduced) = 31.72, 31.24 %

E. Perceptron trained with MSE

ORL:

- Accuracy with iterations 100 = 91.13, 87.5 %
- Accuracy with iteration 10, 100(PCA reduced) = Under 10%

MNIST

- Accuracy with iterations 10, 100 = 91.47, 91.05 %
- Accuracy with iteration 10, 100(PCA reduced) = 31.41, 30.54 %

VI. Conclusion

Based on the tests we can conclude that the nearest neighbor classifier and the nearest sub-class centroid classifier shows the most promising results for both the MNIST, and ORL datasets. However, the full range of capabilities of the perecptron with SGD with hyper parameter tuning is yet to be explored.
Since the performance of 2 dimension PCA reduced dataset is quite poor, its quite difficult to visualize the results of the classification in 2 dimension.

References

[1] The MNIST database http://yann.lecun.com/exdb/mnist/http://yann.lecun.com/exdb/mnist/
[2] ORL(Our Database of Faces) https://paperswithcode.com/dataset/orlhttps://paperswithcode.com/dataset/orl
[3] sci-kit learn library for ML https://scikit-learn.org/stable/https://scikit-learn.org/stable/
[4] The python programming language https://www.python.org/https://www.python.org/
[5] The Matplotlib library https://matplotlib.org/https://matplotlib.org/