SCC361: Artificial Intelligence 20 October 2023

Comparison of Multi-Classification models on the EMNIST dataset.

Introduction

Supervised image classification on the EMNIST dataset is to be carried out. The EMNIST dataset is commonly used with classification and computer vision systems. As part of this project, we will be using truncated data (26,000 images rather than 100,000 items in the original data set) rather than the original number of images. This is due to computational constraints, resource limitations, reducing bias (by using a subset, we can identify potential biases before scaling to entire data set) etc. We will train 4 models and make inferences from the truncated data. Different models have their advantages and disadvantages; therefore, it is important for us to find the most suitable model for inferring data on the EMNIST dataset.

Data and Preparation

Each image is made of 28x28 pixels. These images are stored as 1x748 vectors for ease of processing and analysis. For example: storing the images as 1D arrays can significantly reduce the overall dimensionality of the data, making the computations and training of the models more tractable and quicker . Load the data into MATLAB and save the labels and features (images) in different parameters. Then we convert the image information from the dataset to a double type. We can reshape the image size to 28x28 allowing us to easily view the images or features when required. We employ a vector approach for generating a random 3x4 matrix of images with their respective labels. Next, we save to a Png File.

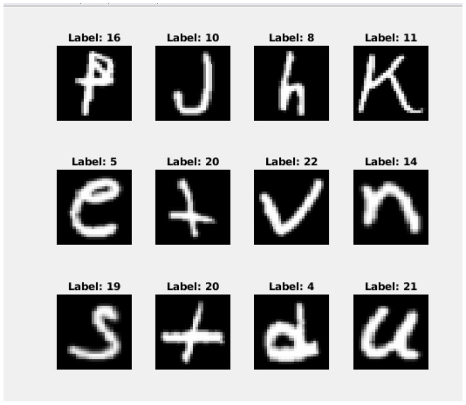
Additionally, we split the data into training and testing data. We do this for both the label and images. We split the training and testing set with a 50% split each. One advantage is: equal representation of training and testing. Furthermore, we have a fair assessment, as the testing set holds enough instances to fairly evaluate the model's performance without being too little or too big. Finally, we check the label distribution for the training and testing data of the labels using the hiscounts function. If the labels are distributed appropriately, this helps us understand the data better.

Methodology

We implement our K-Nearest Neighbour (KNN) model using two different distance metrics (Euclidean distance and L1/Manhattan distance. We define K=5 to be the number of neighbours. To validate our own implementation, we compare our with SVM for multiclass and Ensembles model. This step begins with selecting 2 existing classification algorithms implemented in MATLAB (e.g., SVM Model, Ensembles Model) by using their respective functions (fitcecoc(), fitcensemble(), etc.) available in MATLAB's Statistics and Machine Learning Toolbox.

The format for each model in MATLAB is the model implementation followed by predictions. For example: svmModel, followed by predictSvm. The models are compared based on model performance (training and testing time calculated using tic, toc commands) and accuracy. Comparing these models allows an evaluation of their predictive performance.

We can compare these models for other reasons too. The KNN model is chosen as it is resistant to outliers. The SVM for multiclass model is compared with the KNN model as it has a contrasting approach to classification. Comparing these different approaches helps in understanding their strengths and weaknesses. The Ensembles model can be more complex due to their combination of multiple models, while KNN is relatively simple. Therefore, comparison with these models can provide insight into the relationship between model complexity and performance.



Results

Classification Model	Accuracy (3.dp)	Training Time (3.dp)	Testing Time (3.dp)
KNN (Euclidean)	0.0389	N/A	877.0688
KNN (L1/Manhattan)	0.758	N/A	903.0182
SVM for Multiclass	0.723	49.441	13.565
Ensembles	0.451	123.882	1.0035

The model with the greatest accuracy is the KNN model with L1 implementation, with SVM for multiclass having a similar accuracy rate. The model that had the lowest accuracy rate is KNN implementation of Euclidean; significantly lower than the L1 implementation. Several reasons for this include Euclidean distance being sensitive to outliers.

Additionally, in high dimensional spaces, the distances between points become less meaningful. Therefore, L1 distance may be less affected by this. The KNN training times are so low that it becomes redundant as part of comparison. However, the training time for the L1 distance is slightly greater. This could be due to the algorithmic complexity of the L1 distance in comparison with the Euclidean. The SVM has a training time and testing time that isn't too skewed compared to the other results i.e., not too large, or too small. In comparison, Ensembles has an accuracy that is quite low. This may be due to lack of computational resources. As a result, it may have converged to suboptimal solutions.

Conclusion

Taking into consideration both accuracy and computational time (training time and testing time), the best model out of the three is SVM for multiclass. Furthermore, SVMs are effective for high dimensional data, which can be the case with images. Also, they work well with small-medium sized classification. However, when using SVM for multiclass in other scenarios, it may be difficult to interpret results, especially compared to K-NN model, which is slightly simpler in this regard.

Further research with models that work will with larger datasets may be useful (save resources and time). In the future, when carrying out the K-Nearest Neighbour , it would be more effective to find the optimal value of k with this data in MATLAB using a reasonable range (e.g., 1 to 20).