CSE 574 - Introduction to Machine Learning
Project 3 Report

Ajith Kumar Natarajan
UB Person number: 50318505
Fall 2019
ajithkum@buffalo.edu

Abstract

In this project I have made a study on various unsupervised learning methods. The dataset used throughout the project is the *Fashion-MNIST* images. The three different types of unsupervised clustering implementations in this work are: K-means; K-means on condensed representation of data obtained through the middle layer of autoencoder (encoder part) and Gaussian Mixture Model (GMM) on condensed representation of data obtained through the middle layer of autoencoder (encoder part).
It is observed that the accuracy percentage increases with the implementation of autoencoder. Also, GMM proves to be more accuracate when compared to K-means. The results obtained can be summarised as follows:

- Implementation of baseline K-means gave an accuracy of 50.02%
- After implementing an autoencoder consisting of 3 convolution layers, the output of the encoder layer was used to build clustering model on the condensed representation. Implementation of K-means on this representation produced an accuracy of 59.55%
- Implementation of K-means on this representation produced an accuracy of 63.24%

1 Introduction

Unsupervised learning is one of the commonly used techniques of machine learning. It is widely used in recommender systems.

In this work, different types of clustering are attempted in fashion-MNIST dataset. Since the number of features (and hence the number of dimensions) on which the clustering is to be done is high, the performance of the clustering technique is not very good. Implementation of dimentionality reduction technique like autoencoder to extract and use the condensed representation helps in increasing the accuracy.

2 Dataset

The Fashion-MNIST is a dataset of Zalando's article images, consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Hence the number of features available to train the model is 784. This feature set consists of single pixel-value associated with each fashion apparel, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The classes are:

- T-shirt/top
- Trouser
- Pullover
- Dress
- Coat
- Sandal
- Shirt
- Sneaker
- Bag
- Ankle Boot

3 Pre-processing

The dataset being read is normalized. This is a common step in almost all the machine learning model building exercise. In order to avoid the effect of one particular column, having high values from inhibiting the other features and to prevent the issue of exploding gradients, all the columns have been normalised so that the value of the columns are in the range 0-1 consistently.
The dataset is also split into training and validation datasets to train the model and verify using the validation data that the model is not overfitting or underfitting.

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

4 Architecture

4.1 Task 1: Baseline K-means

There is no special architecture for baseline k-means on the data. The initial centroids are chosen using the k-means++ implementation. This method ensures that the chosen clusters have high inter-cluster distance when being initialized.

4.2 Task 2 and Task 3: K-means & GMM on condensed representation of encoder

For the tasks 2 and 3, an autoencoder is designed. After training the autoencoder, the encoder is separated. This encoder gives the condensed representation of the data it is being fed into. K-means (task 2) and GMM (task 3) models are fit on this representation.

5 Results

This section consists of the results obtained by implementing the models mentioned above.

5.1 Task 1: Baseline K-means

The normalized input vector is clustered with different number of clusters to observe the ”elbow” graph. The ”elbow” point from the graph is chosen to be the number of clusters.

With K-means being an unsupervised learning method, even though the model learns to cluster the set of data, the predicted cluster labels do not match with the original annotated data labels. Hence, the confusion matrix generated seems to have a poor accuracy.

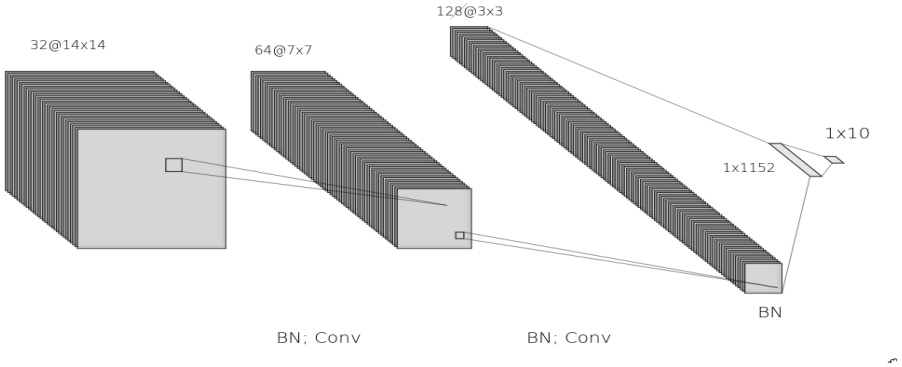

Figure 1: Architecture diagram of encoder used for task 2 and task 3

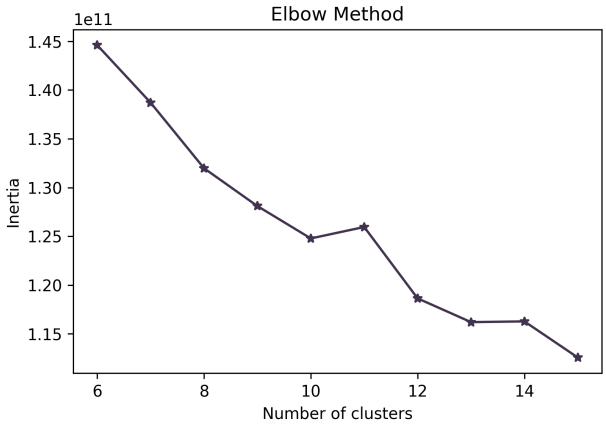

Figure 2: Basis for choosing the number of clusters

To fix this, the labels predicted by the model is iteratively reassigned such that the maximum value of the prediction matches with the expected label. This was implemented using Hungarian algorithm to optimize the operation. The confusion matrix obtained after this process is shown here:

The accuracy of the model, which can be calculated from the confusion matrix as the division between the number of correctly classified sample and the total number of sample is **50.02%**.

5.2 Task 2 and Task 3: K-means & GMM on condensed representation of encoder

This is the graph depicting the variation of training and validation loss as a function of epochs while training the model. It is observed that both the training as well as validation loss decreases with increase in epoch - hence it can be concluded that the model is not overfitting or underfitting.

5.3 Task 2 confusion matrices

The confusion matrix obtained before reordering is:

After reordering using the same method detailed above the confusion matrix obtained is:

The accuracy of the model obtained using this architecture for encoder along with k-means is **59.55%**.

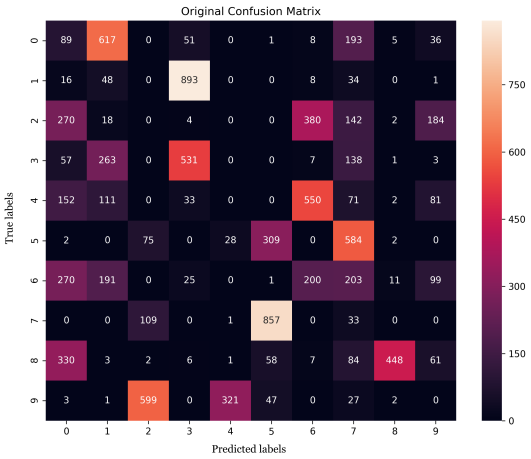

Figure 3: Confusion matrix for task 1 (before reordering)

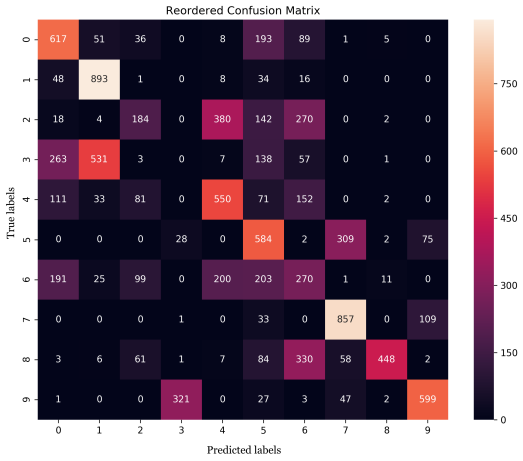

Figure 4: Confusion matrix for task 1 (after reordering)

5.4 Task 3 confusion matrices

The confusion matrix obtained before reordering is:

After reordering using the same method detailed above the confusion matrix obtained is:

The accuracy of the model obtained using this architecture for encoder along with GMM is **63.24%**.

6 Conclusion

Through this work, it was observed that GMM is a better way of clustering than k-means. K-means suffers from two main disadvantages - the clusters are always circular/spherical in shape and that k-means result in hard-clustering. GMM in addition to providing probabilistic result for clustering also enables the clusters to be of Gaussian shapes. Another observation that can be made is that the clustering suffers from the curse of dimensionality and hence it is not so reliable to use directly

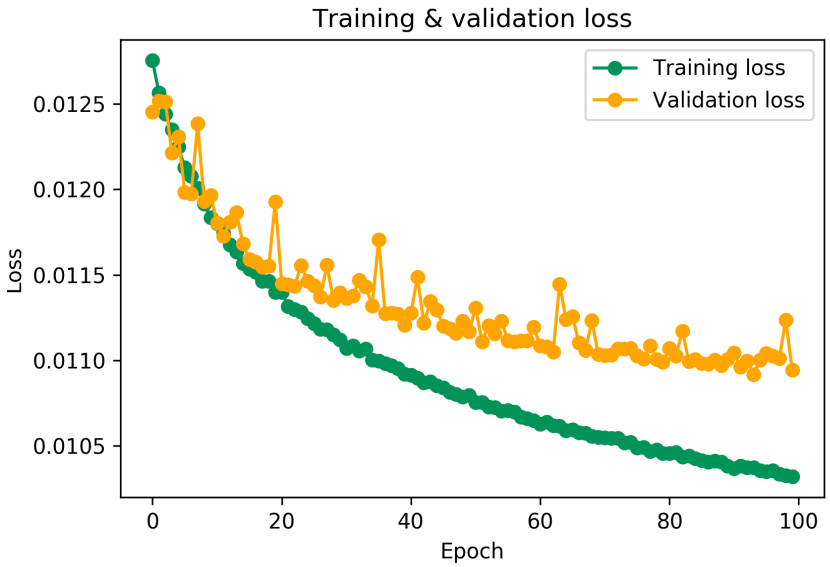

Figure 5: Training & validation loss plotted against number of epochs

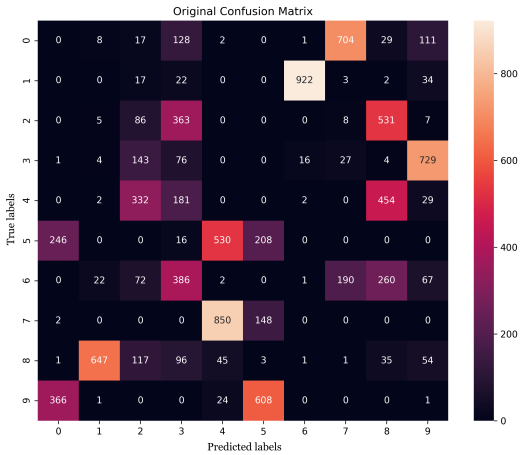

Figure 6: Confusion matrix for task 2 (before reordering)

when the number of dimensions is too high. Using dimensionality reduction technique such as auto-encoder's encoder helps in increasing the accuracy.

References

[1] Bishop, Christopher M. Pattern Recognition and Machine Learning. New York :Springer, 2006

[2] Srihari, Sargur N. Lecture Slides for Machine Learning

[3] Stack Overflow

[4] VanderPlas, Jake: Python Data Science Handbook

[5] Tensorflow documentation

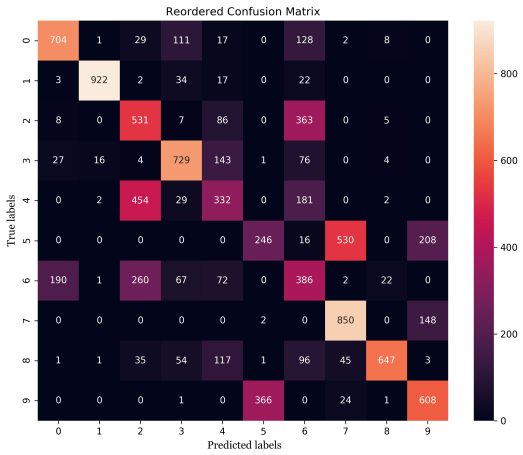

Figure 7: Confusion matrix for task 2 (after reordering)

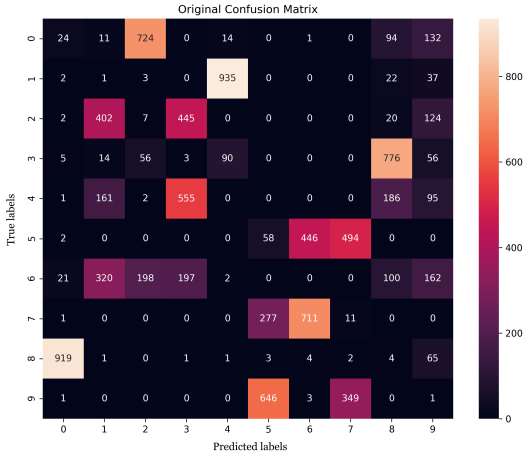

Figure 8: Confusion matrix for task 3 (before reordering)

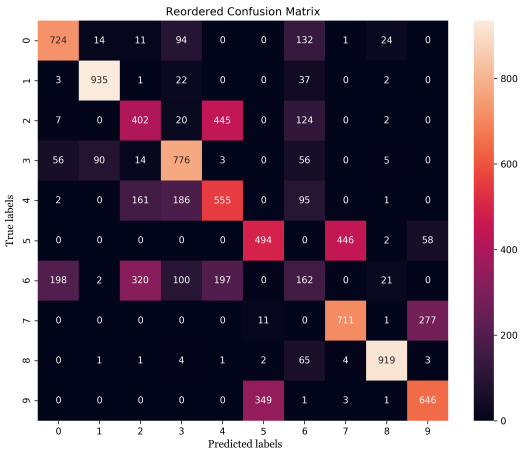

Figure 9: Confusion matrix for task 3 (after reordering)