Playing around FashionMNIST dataset

Alessi Michele

May 2023

Abstract

This project aims to explore the FashionMNIST [1] dataset using a combination of unsupervised and supervised learning techniques. The first phase involves a geometry analysis to reduce the dataset's dimensionality, followed by an unsupervised phase to classify the reduced-data. The unsupervised phase will provide labels that will be used in the supervised learning phase. During the latter, the training data will be used along with the labels coming from unsupervised phase to train the model: this part will be addressed as semi-supervised approach. Finally, a fully-supervised learning approach will be performed using the true training labels. The results obtained from semi-supervised and fully-supervised approaches will be compared to each other. The ultimate goal of the project is to evaluate the effectiveness of the semi-supervised and fully supervised approaches in the classification task. The code utilized can be found here [2].

1 Load data

After the data was loaded, 1000 training data and 300 test data were sampled from the complete dataset to reduce computation time. The data was then normalized and exported into NumPy arrays for geometric analysis and unsupervised part. To perform stochastic gradient descent, a DataLoader from PyTorch was utilized for training the neural networks. This allowed efficient batch loading and shuffling of the data during the training process. Figure 1 displayed the first ten images from the training set, along with their labels to give an overview.

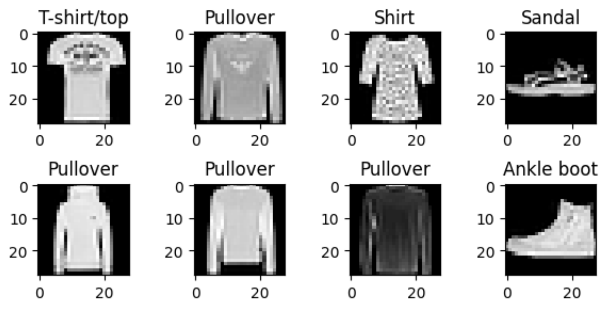

Figure 1: First 10 training images with labels

2 Understanding data geometry

The analysis of the underlying geometry of the data involved three approaches namely Linear PCA [3] (figure 2), Kernel PCA [4] (figure 3) with an rbf kernel (figure 4), and Kernel PCA with a polynomial kernel. A plot was created using these approaches to visualize the results.

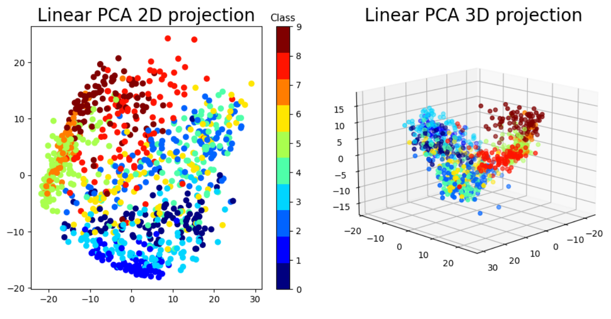

Figure 2: Linear PCA

From the plot, it was observed that a gamma value greater than 0.1 for rbf kernel did not provide a good projection. However, for gamma values close to zero, the projection is effective in understanding the underlying data geometry. This observation allowed a clear separation between "brighter" and "darker" labels. For the polynomial kernel, it was observed that as the degree increased, the data became more stretched, making it difficult to understand the geometry. On focusing on lower values (up to 5), a pretty good separation was achieved, but there was still some problems. For instance, the green-scale and light-blue scale points were spread out and mixed each other, making it challenging to differentiate them.

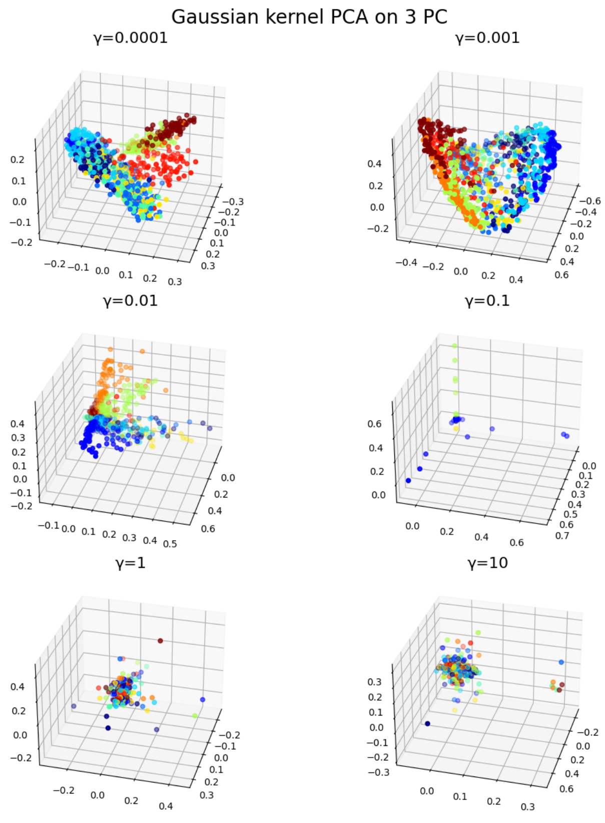

Figure 3: Gaussian kernel PCA

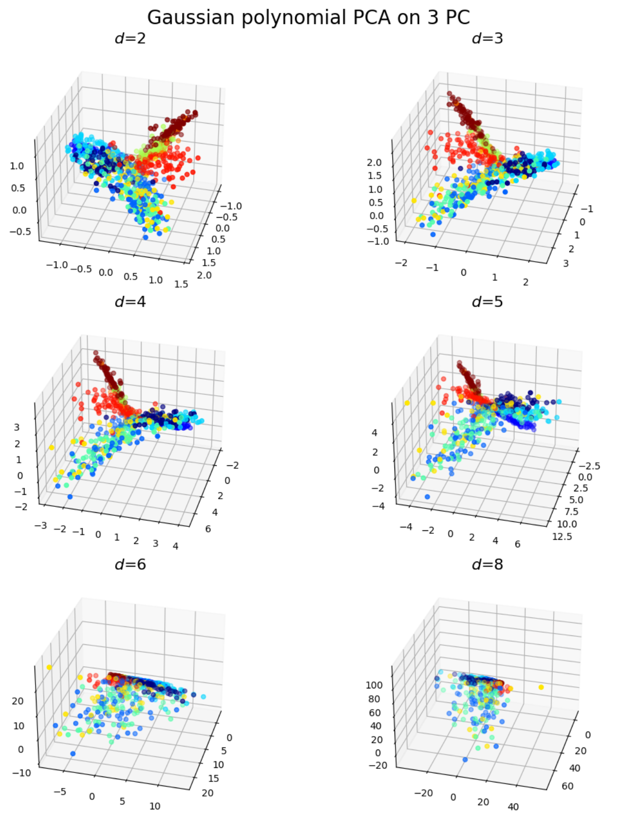

Figure 4: Polynomial kernel PCA

3 Bridging unsupervised and supervised

In this section, the method to best separate the data geometry was selected, and it was used to project the data onto its 10 principal components. The resulting reduced-dimension dataset was then subjected to clustering, where each image is assigned a label ranging from 0 to 9.

Following the previous analysis, the rbf kernel was found to be better than the polynomial kernel. The optimal gamma values were identified to be 0.0001 and 0.001 by visualizing the projection. Figure 5 shows an example of the projection of the same images in 1 onto their 10 principal components using the rbf kernel with gamma equal to 0.0001.

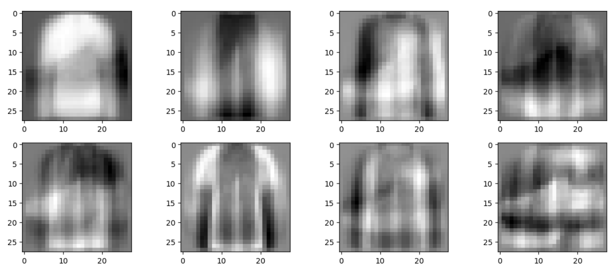
Figure 5: Projection of the first 10 training images

Subsequently, unsupervised learning was introduced, and kmeans [5] was determined to be a suitable option. To evaluate the projection and clustering, the correspondence between the kmeans algorithm-assigned labels and the ground-truth labels was assessed. It is important to note that the labels assigned by the kmeans algorithm might not match the true labels (*kmeans might permute the labels of the classes*). To address this issue, the homogeneity score [6] and the completeness score [6] were used to evaluate performance, taking into account the aforementioned permutations. Figure 6 displays two confusion matrices.

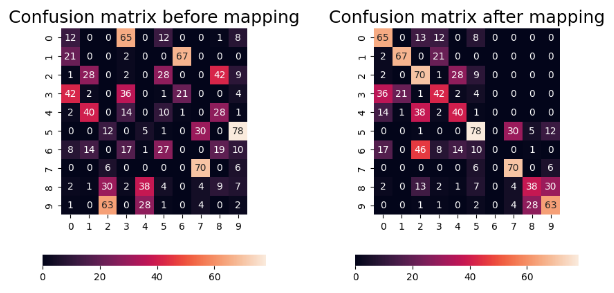

Figure 6: Confusion matrix

The first confusion matrix compares the ground-truth labels with the labels generated directly from the kmeans algorithm. On the other hand, the second confusion matrix compares the ground-truth labels with the kmeans mapped-labels. In this case, kmeans clusters are mapped to the true label that is more frequent within that cluster. Note that homogeneity score and completeness score don't change if computed with the mapped-labels or with the unmapped-labels, since those metrics are desgined to be *names-independent*. It was found that homogeneity score and completeness score were 48.2% and 48.6%, respectively. Overall, these confusion matrices help to evaluate the performance of the projection and clustering methods used in the analysis: based on this, it can be concluded that while the accuracy is not exemplary, the combination of dimensionality reduction with kmeans clustering has led to satisfactory results.

Furthermore, figure 7 shows a plot of the eigenvalues spectrum generated by the dimensionality reduction method chosen (rbf kernel). This plot is required to confirm that the choice of projecting onto 10 principal components is consistent with the so called elbow rule.

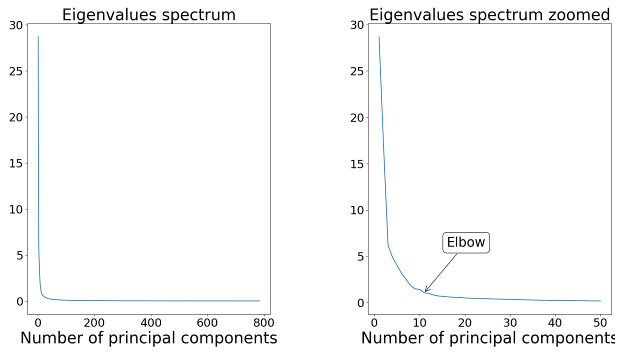

Figure 7: Eigenvalues spectrum

4 Classification

After unsupervised part, four classifiers were trained [2]. Firstly, the labels generated from the kmeans clustering algorithm were utilized without considering the actual matching with the true labels. Then, the same classifiers were trained using the true labels. The four classifiers were Support Vector Machine (SVM) [7] with radial basis function kernel (rbf), two Fully Connected Neural Network (FCNN) [8] with different architecture, and Convolutional Neural Network (CNN) [8]. The SVM was chosen after performing a grid-search cross validation: the best hyper-

Table 1: Test Accuracy

Model	Accuracy on kmeans labels before map.	Aaccuracy on kmeans labels after map.	Accuracy on mapped kmeans labels	Accuracy on ground-truth labels
SVM	22.6%	53.5%	53.3%	87.5%
FCNN1	22.6%	53.8%	54.1%	84.57%
FCNN2	10.5%	11.3%	10.5%	9.7%
CNN	22.9%	53.2%	40.8%	85.8%

parameters turned out to be $C = 10.0$ and gamma value 0.001. For the FCNN, the first had one hidden layer with 128 neurons, while the second had one hidden layer with 256 neurons, both followed by a leaky ReLU activation function. The CNN had three convolution layer with 16, 32 and 64 output channels respectively. Each convolutional layer has been combined with average pooling layers with 2x2 kernel size, followed by a leaky ReLU activation function.
In the first part of the classification task, the aim was to classify the data points into their corresponding clusters as determined by the kmeans algorithm, without any knowledge of the actual ground-truth labels. In the second part, the goal was to compare the effectiveness of these two approaches.

5 Assess the pipeline

In this section, the overall accuracy of the classifiers on the test set was evaluated by comparing the predicted labels with the ground-truth test labels. It should be noted that for the first method (using labels from kmeans clustering algorithm), the predicted labels may not match the ground-truth labels. To address this, the same approach as the one used for building the mapped version of the labels generated by kmeans was employed. The predicted labels is grouped, and each group is mapped to the ground-truth label that is most represented in that group. Evaluating the accuracy of the second method (using ground-truth labels) is instead straightforward.

Two distinct training approaches were utilized so far. The first method involved training using kmeans-generated *unmapped* labels, followed by label *reordering/mapping* after the training. The second was training using the readily available ground-truth labels. A third approach was tested in [2]: directly train the classifier with mapped labels generated by kmeans (i.e. reordering of the labels is performed before training). This approach yielded similar results as the first method due to the inconsequential nature of label naming. Kmeans recognizes the intrinsic characteristics, making mapping the labels before training or training with unsorted labels and reordering after producing relatively equal results. All the results can be found in table 1. Note that the test accuracy is roughly the same for SVM, FCNN1 and CNN, while FCNN2 has significantly lower test accuracy: this could be due to the fact that FCNN2 has higher number of parameters (two times the neurons of FCNN1), and it is well-known that many degrees of freedom lead to overfitting: probabily this cause low results for FCNN2 on the test set.

6 Conclusions

In conclusion, this project employed a combination of unsupervised and supervised learning techniques to classify the FashionMNIST dataset. Three dimensionality reduction methods were analyzed; then the reduced-dimension dataset was clustered using kmeans and the labels produced were used for semi-supervised and supervised learning tasks. SVM, FCNN, and CNN models were trained and their accuracy was subsequently evaluated: the results show that the semi-supervised approach using kmeans-generated labels can yield to satisfactory results. Overall, this study demonstrates the effectiveness of unsupervised learning as a precursor to supervised learning approaches, and it also highlights the intrinsic ability of unsupervised learning techniques to recognize data features, regardless of label names.

Figure 8 displays performance of SVM on the same 8 images in the three different approaches: training on *pure* unmapped labels, training on unmapped labels but with labels-reordering after training, training directly on mapped labels, training on true labels.

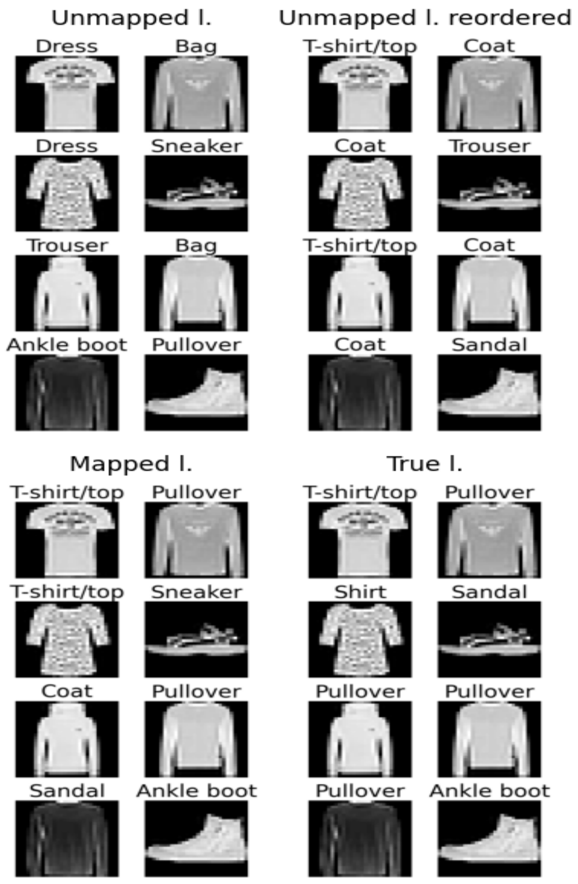

Figure 8: Comparison of different approaches.

References

[1] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.

[2] M. Alessi, "Fashionmnist," 2023. `https://github.com/alessimichele/FashionMNIST.git`.

[3] K. P. F.R.S., "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[4] B. Schölkopf, A. J. Smola, and K.-R. Müller, *Kernel Principal Component Analysis*, p. 327–352. Cambridge, MA, USA: MIT Press, 1999.

[5] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. `https://scikit-learn.org/stable/modules/clustering.html#homogeneity-completeness-and-v-measure`.

[7] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[8] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* Cambridge, MA, USA: MIT Press, 2016. `http://www.deeplearningbook.org`.