Advanced Deep Learning and Kernel Methods
Challenge 1

Valentinis Alessio
Università degli Studi di Trieste

1 Introduction

This report will present the results of the first challenge in the Advanced Deep Learning and Kernel Methods course. The challenge consists in developing a Machine Learning pipeline for the Fashion-MNIST dataset, which exploits both Unsupervised and Supervised learning techniques. The pipeline was composed of the following steps:

- Preprocessing
- Dimensionality Reduction
- Clustering
- Classification

2 Source Code

The complete code, including all experiments and implementation discussed in the following sections is available at the following GitHub repository.

3 Dataset

The dataset used for this challenge is the Fashion-MNIST dataset, available as Torch Dataset. It consists of 60000 black and white images of size 28×28 pixels, divided into 10 classes. So, the dataset is composed of 784 features. I opted for a simple z-score normalization of the data, in order to have zero mean and unit variance. However, some modifications had to be made in order to avoid division by zero, so I added a small constant to the variance. Moreover, if we look at the very corners of the images, we can see that values are always zero, so even adding a small constant to the variance, the normalization would end up in huge values in the corners: to avoid this issue I decided to clip the values that are greater than 100, due to too little variance, to 0.

4 Dimensionality Reduction

The first step of the challenge consisted in a dimensionality reduction phase, applying both PCA and kernel-PCA.

4.1 Principal Component Analysis (PCA)

First I applied linear PCA to teh whole dataset and visualized the first two and three principal components. The results are shown in Figure 1.

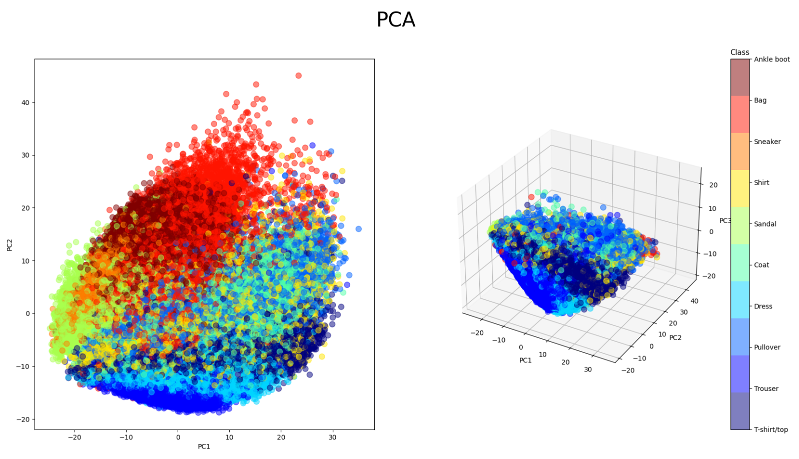

Figura 1: PCA visualization of the first two and three principal components.

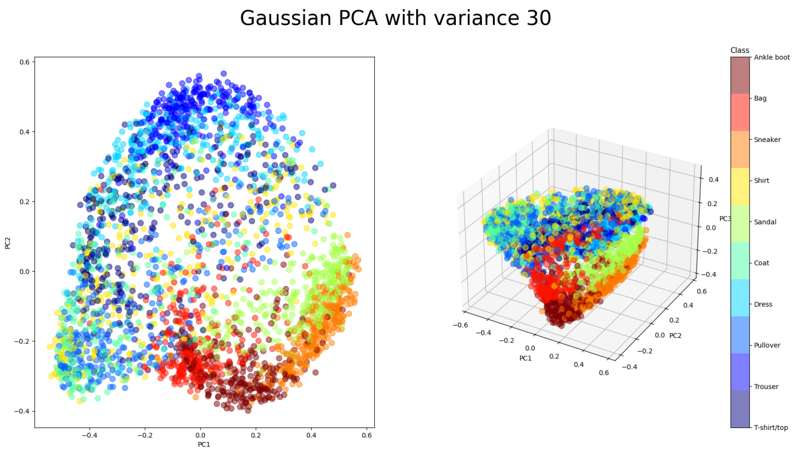

Figura 2: Kernel PCA visualization of the first two and three principal components.

4.2 Kernel PCA

I then went on to apply Kernel PCA to the dataset, experimenting with different kernels and parameters using both RBF, polynomial and sigmoid kernels. The algorithm was applied to $\frac{1}{20}$-th of the dataset, due to computational reasons, however the results are still meaningful.

The RBF kernel, with variance parameter $\sigma = 30$ provides the best separation, capturing the non-linear relationships between the data points without incurring in overfitting. The results are shown in Figure 2.

5 *Hybrid* Unsupervised - Supervised Approach

5.1 KMeans Clustering

Then I proceeded to first reduce the dimensionality of the dataset using Kernel PCA with RBF kernel and $\sigma = 30$ on 10 components. As pictured in the Figure 3, we can picture an elbow in the spectrum of the explained variance, which suggests that 10 components are enough, even though the cumulative explained variance is not very high. This phenomenon is expectable, as we are trying to shrink the dataset from 784 to 10 dimensions.

Then I applied KMeans clustering to the reduced dataset, with $k = 10$ clusters, in order to see if the obtained clusters reflect the original label distribution. The results are shown in Figure 4. Looking at the label distribution in the clusters, we can see how most of them are composed of mainly one

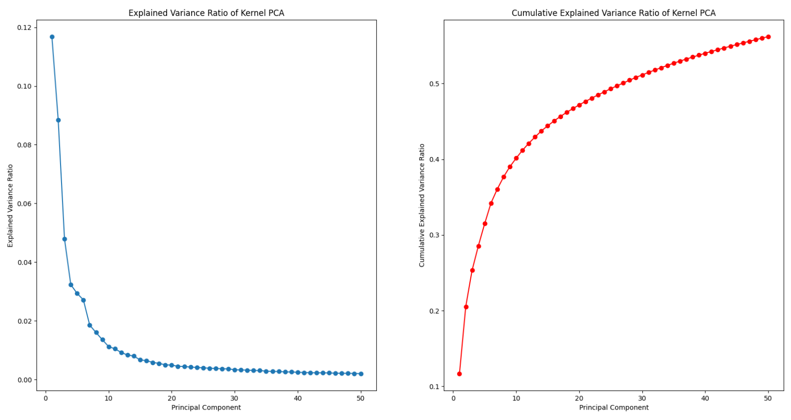

Figura 3: Explained variance spectrum of Kernel PCA with RBF kernel and $\sigma = 30$.

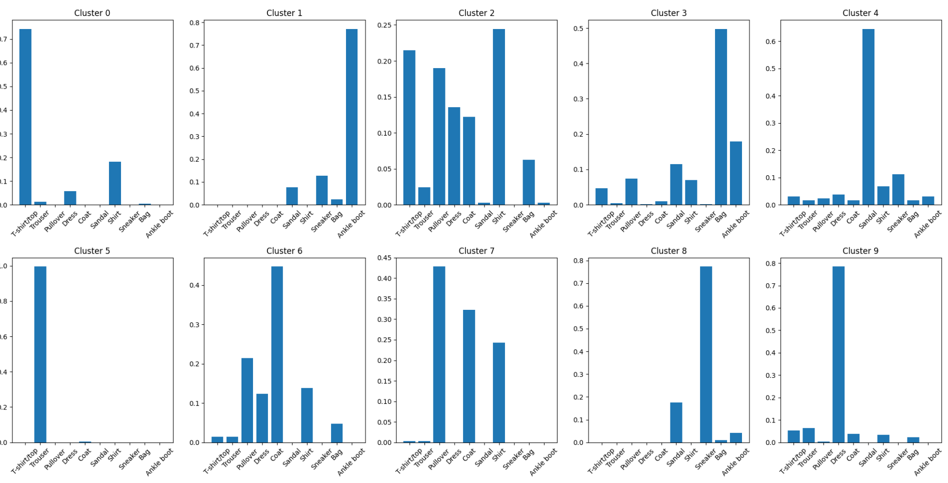

Figura 4: KMeans clustering of the reduced dataset.

class, reflecting a good separation between classes. However, some of them are characterized by a mixed distribution of about 2-3 classes, and mainly referred to upper-body clothes.

5.2 Supervised Classification on Clusters

Finally, I applied a supervised classification algorithm to the clusters obtained from the KMeans clustering. I used various models, such as kernel-SVM and Neural Networks. In what follows, I will briefly recap the performances of the models trained. The best models were selected based on the validation accuracy.

5.2.1 Kernel-SVM

For this part I experimented with various kernels, such as RBF, polynomial and sigmoid, trying to perform also some hyper-parameter tuning.

- Gaussian RBF Kernel:

  $\sigma = 10$ shows clear signs of overfitting, with a training accuracy of 1.0 and a validation accuracy of 0.73.

  Best Model: $\sigma = 25$, with a training accuracy of 0.95 and a validation accuracy of 0.93.

- Polynomial Kernel:

  This kernel showed more consistent performance, both on training and validation accuracy.

Best Model: $d = 3$, with a training accuracy of 0.92 and a validation accuracy of 0.92.

- Sigmoid Kernel:

  This kernel showed the worst performance overall, with very poor validation accuracy.

  Best Model: $\gamma = 0.01$, with a training accuracy of 0.41 and a validation accuracy of 0.43.

Overall, the best model resulted to be the one exploiting the Gaussian RBF kernel, with $\sigma = 25$, with both high training and validation accuracy, symptom of a good generalization.

5.2.2 Fully Connected Neural Networks

I also experimented with fully connected neural networks, with different architectures and hyperparameters.

- Smaller Neural Network:

  128 hidden units for two layers, with ReLU activation function, 20 epochs and learning rate of 0.001.

  Reached a training accuracy of 1.0 and a validation accuracy of 0.92.

  Shows signs of overfitting at later epochs.

- Bigger Neural Network:

  256 hidden units for two layers, with ReLU activation function, 20 epochs and learning rate of 0.0005.

  Reached a training accuracy of 1.0 and a validation accuracy of 0.93.

  Slight better gereralization roperties than the first network, but still shows signs of overfitting at later epochs.

5.2.3 Convolutional Neural Networks

Finally, I experimented with Convolutional Neural Networks, which are known to perform well on image classification tasks.

- Achieved almost the best performance among the models trained, with a validation accuracy of 0.93.
- Keeping the training accuracy lower than 1.0, we can see that the model is able to generalize better.
- Showed lower signs of overfitting than the fully connected neural networks.

5.2.4 Comparisons

1. Kernel-SVM outperformed all types of Neural Networks, with the best model reaching a validation accuracy of 0.94.
2. Among the neural networks, the Convolutional Neural Network showed the better generalization properties, likely to their ability to capture spatial relationships in the data.
3. Both FCN showed signs of overfitting, above all when trained on higher number of epochs.

These results suggest that the best model for classifying the clusters obtained from the KMeans clustering on reduced data is the kernel-SVM, maybe due to the fact that the clusters were constructed with kernel-reduced data on similar parameters. However, the Convolutional Neural Network results show that on further optimization (even simply more data), they could match or even outperform the kernel-SVM.

High validation accuracies on cluster labels shows that clustering was able to capture the main underlying structure of the data. It is important to note, however, that this classification is made on the clustering labels, not on the actual Fashion-MNIST labels, which can explain the strangely high accuracies in a dataset like this, where the usual accuracies are around 0.9.

5.3 Performance on Test Set with real labels

As final part of the pipeline, I tested the best model obtained on the test set, with the real labels. In order to conduct this test, I employed two methods: one consisting in assigning as *real label* of each cluster the most frequent label in the cluster (this approach is called *Majority Voting*, which could be mimicked by sampling some points from each cluster and manually assign the label related to the most frequent class), and another one more *probabilistic* in a fuzzy c-means fashion, where the label of a point is assigned sampling from the probability distribution of the labels in the cluster.

The resulted about class-wise and weighted accuracies are reported in the table below.

Class	**SVM**	**FCN1**	**FCN2**	**CNN**
T-shirt/top	0.58	0.58	0.58	0.59
Trousers	0.91	0.90	0.90	0.90
Pullover	0.41	0.43	0.43	0.42
Dress	0.68	0.67	0.67	0.67
Coat	0.47	0.47	0.47	0.47
Sandal	0.55	0.52	0.52	0.53
Shirt	0.27	0.27	0.27	0.26
Sneaker	0.79	0.78	0.78	0.78
Bag	0.60	0.60	0.60	0.60
Ankle boot	0.72	0.71	0.71	0.71
Accuracy	0.59	0.59	0.59	0.59

Tabella 1: Class-wise and weighted accuracies on the test set using majority voting.

Class	**SVM**	**FCN1**	**FCN2**	**CNN**
T-shirt/top	0.38	0.40	0.39	0.40
Trousers	0.84	0.83	0.83	0.83
Pullover	0.30	0.29	0.27	0.29
Dress	0.51	0.53	0.50	0.50
Coat	0.37	0.35	0.35	0.36
Sandal	0.33	0.33	0.36	0.33
Shirt	0.20	0.17	0.18	0.20
Sneaker	0.65	0.66	0.63	0.63
Bag	0.39	0.41	0.42	0.41
Ankle boot	0.58	0.59	0.58	0.58
Accuracy	0.45	0.46	0.45	0.45

Tabella 2: Class-wise and weighted accuracies on the test set using fuzzy assignment.

Table 1 shows the results using the majority voting method, while Table 2 shows the results of the test set evaluation using the fuzzy assignment method,. We can observe that the results are in both cases pretty low, reaching some consistent 60% accuracy in the *majority voting* scenario, while only about 45% in the fuzzy assignment. This is due to the fact that the clusters obtained from the KMeans don't mirror exactly the original labels, with more clusters showing mixed label distributions. This is reflected even more in the accuracy of the fuzzy assignment, as introducing a probabilistic assignment makes the model more sensible to the mixed distributions in the clusters.

However, we can see some interesting results regarding the class-wise accuracies:

- Trousers, Sneakers and Ankle boots are the classes with the highest accuracies, showing that the clusters obtained for these classes are *cleaner*, in the sense that they contain mainly points of the same class.
- Looking at the *majority voting* accuracies, we have "high" accuracies also for the labels Dress and Bag.
- The class with the lowest accuracies are Shirt, Coat and Pullover, which are the classes with the most mixed distributions in the clusters. This behavior can be explained by the fact that these classes are more similar to each other, so we expect that characterizing features lie in higher

dimensions than the ones used for the clustering (10 dimensions representing about 40% of the variance of the data).

This result highlights how the unsupervised-to-supervised pipeline captured some maningful structure in the data, reflected in the high validation accuracies, but there is a significant drop in performance when we pass to the original Fashion-MNIST labels. The consistent difficulty of the models to distinguish classes like T-shirt/top and Shirt, or Coat and Pullover, shows how these classes were mixed during the clustering phase, maybe due to the fact that they share some features in the reduced space. Maybe moving to a higher-dimensional space above all when using the kernel-PCA could help in capturing more complex relationships between features and help in distinguishing these classes during clustering phase.

The significantly low performance of these models (45-60% accuracy) on the test set on this type of dataset, compared to the usual accuracies classification accuracies on Fashion-MNIST (above 90%), highlights significant challenges in the unsupervised-to-supervised pipeline, especially when dealing with mixed distributions in the clusters. However, the fact that all models perform well above random guessing (10% accuracy) shows that the pipeline was able to capture some meaningful structure in the data.

6 Fully Supervised Approach

As last step, I performed the classification approach in a fully supervised framework, so using the true label assignment during training.

6.1 Model Performance

I trained the same models used in the previous section, but this time using the true labels for training. The results are shown in the table below.

Model	Hybrid pipeline (fuzzy assignment)	Fully supervised
SVM	0.45	0.81
FCN1	0.46	0.55
FCN2	0.45	0.59
CNN	0.45	0.66

Tabella 3: Comparison of the models' performance in the hybrid and fully supervised pipeline.

6.2 Class-wise Accuracies

The class-wise accuracies of the models trained in the fully supervised pipeline are by far higher than the ones obtained in the hybrid pipeline, showing a consistent value above 60% for almost all classes, but again the classes Shirt, Coat and Pullover show the lowest accuracies, of around 40%.

This result highlights how the fully supervised approach is able to capture more complex relationships between features and labels, and is able to distinguish more easily between classes that are more similar to each other.

6.3 Trial of Neural Networks on not-scaled data

As a last attempt, I tried to train the Neural Networks on the not-scaled data, in order to see how the models are able to deal with the scaling of the data. The results are shown in the table below.

Model	Scaled data	Not-scaled data
SVM	0.81	-
FCN1	0.55	0.81
FCN2	0.59	0.81
CNN	0.66	0.86

Tabella 4: Comparison of the models' performance on scaled and not-scaled data.

The results show that the Neural Networks are able to deal autonomously with the scaling of the data, by learning the proper one. In this framework we can see how the Fully Connected Neural Networks are able to perform as the SVM, while the Convolutional Neural Network is able to perform even better, reaching an accuracy of 0.86 on the not-scaled data.

7 Discussion

7.1 Dimensionality Reduction and Clustering

The PCA and Kernel-PCA highlighted the complex non linear relations in the Fashion-MNIST dataset. The k-Means clustering on the reduced data was able to capture some meningful insights in the data, as showed by the distribution of labels in each cluster. However exploiting only these few dimensions (10) for clustering resulted to be not enough to capture the full complexity of the data, as showed by the repetition of the most frequent label per cluster, and the presence of mixed distributions in some clusters.

7.2 Hybrid vs. Supervised Classification

The hybrid pipeline showed some interesting results, demonstrating some potential in employing some preliminary unsupervised learning techniques to create a rudimental *baseline* classification system. However, the substantial performance gap with respect to the fully supervised approach (of about 10-40%) shows how the importance of label assignment in this task.

Moreover, the performance gap of the NNs on the not-scaled data, compared to the scaled data, shows how powerful the Neural Networks are in not only capturing the meaningful structure in the data, but also in dealing with the proper transformations.

7.3 Model Comparison

Unexpectedly, applying the kernel-SVM and the NNs on the same data, showed how in principle simpler models like the kernel-SVM are able to outperform the NNs, outlining how the decision boundaries on this kind of data can be easily captured by SVM's kernel trick. However, we can see how, feeding the Neural Networks with not scaled data we are able to reach better performances, even on a small amount of data.

8 Conclusions

This challenge showed how we can deep-dive into Kernel methods and Neural Networks using only a toy dataset like Fashion-MNIST. The results showed, expectedly, how a fully-supervised approach performs by far better than a hybrid unsupervised-to-supervised pipeline, but how the latter is able to provide some meaningful insights in the data structure. Moreover, seeing how SVMs are able to perform almost equally to fully connected Neural Networks in this image classification task shows how powerful the kernel trick can be in capturing complex relationships in the data, above all when dealing with small datasets.