Zhaoyan Lin
Professor Georges Grinstein
CSC 390
Notebook: https://github.com/youtuyy/CSC390/blob/master/CSC390/fashionmnist/Project.ipynb
Data: https://www.kaggle.com/zalando-research/fashionmnist/data
21 December 2017

Fashion Item Classifier: Visualization of Different Models

1. INTRODUCTION

The original MNIST database of handwritten digits is a popular database widely used to benchmark machine learning algorithms. With MNIST dataset, people who want to try their learning algorithms can minimize their time of preprocessing data. The european electronic commerce company Zalando now tends to replace MNIST with its Fashion MNIST dataset of its articles' images. This project applies three different supervised models, including logistic regression, Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN), to classify these clothing articles. Another focus of this project is to visualize the learning techniques and look into how these algorithms learn from the images.

2. DATA

The Fashion MNIST dataset is a MNIST like dataset. It consists of 60,000 training examples and 10,000 testing example. Each example is a 28*28 grey scale image associated with one of the ten classes (T-shirt/ top, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot). The dataset is in a csv format with 785 columns. Each pixel is an integer value from 0 to 255, showing the darkness.

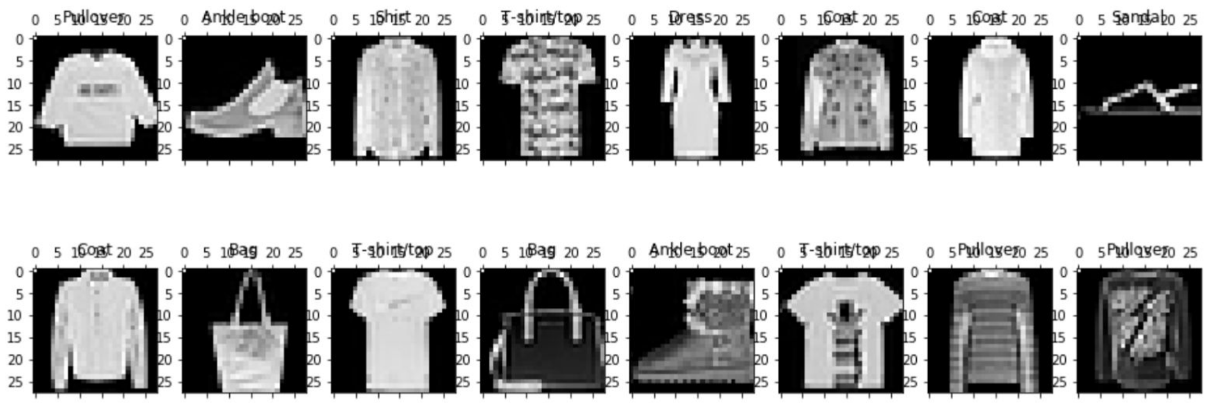

Figure 1.

The dataset is really well processed, so it saved a lot of time to deal with data and the project could focus on the implementation of more algorithms.

2.1 DATA VISUALIZATION

First, I choose to use the most widely used dimensionality reduction method Principal Component Analysis (PCA) for the visualization. Figure 2 shows the result trying to reduce the dimension to 2 with PCA. There are overlaps everywhere. PCA does not seem to work very well in this case.

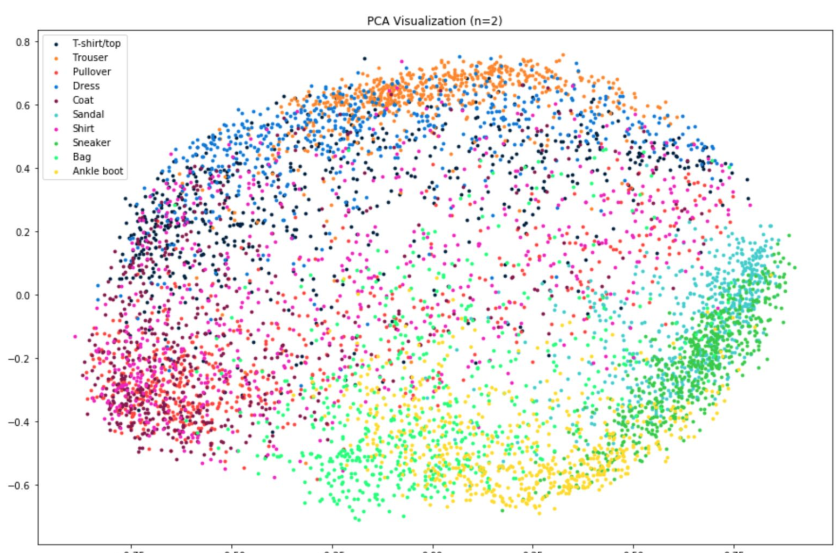

Figure 2.

Thus, I turn to another method called t-SNE, which is a relatively newly developed dimensionality reduction technique. The result is much better than PCA. Although there are still overlaps in the categories shirt, pullover and coat, most of the other categories are separately distributed on the graph. I note that two classes, trouser and bag, hardly overlap with other classes.

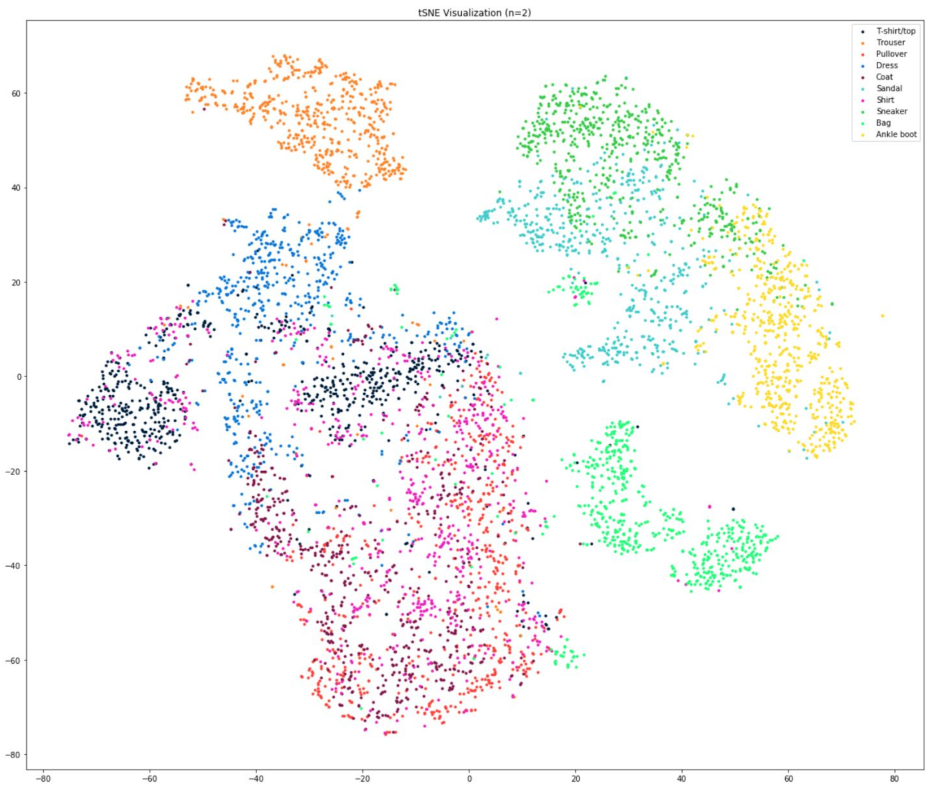

Figure 3.

3. TRAINING MODELS

After visualizing the data, I move on to training different models. Besides, I look at the features learned by the model and try to understand the learning behavior of the algorithms.

3.1. LOGISTIC REGRESSION

The first model I use is logistic regression with L1 penalty. The classification runs for 17 seconds and ends up with a test score of 77%. The visualization of the classification vectors is shown in Figure 4, which indicates that the model is learning the edge of the items. For the original MNIST dataset, logistic regression can reach an accuracy of 90%. Compared to handwritten digits, the patterns of clothing items are similar and harder to find.

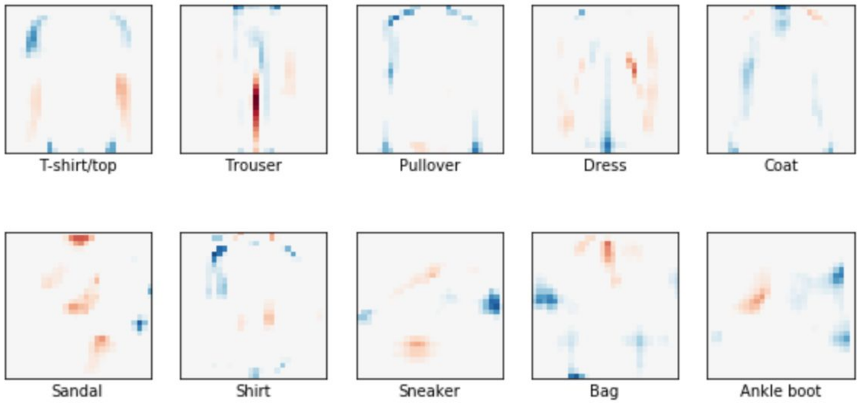

Figure 4.

3.2. MLP CLASSIFIER

Secondly, I implement an MLP classifier. To run it faster, I use very few hidden units at first. The training stops at 26th iteration for that the training loss does not improve more than the tolerance for two consecutive iterations. The test set score is 86%. A visualization of weights of the first layer is shown in Figure 6. It vaguely shows the pattern of the clothing items. The shoulders and the sleeves of the clothes are the most obvious parts.

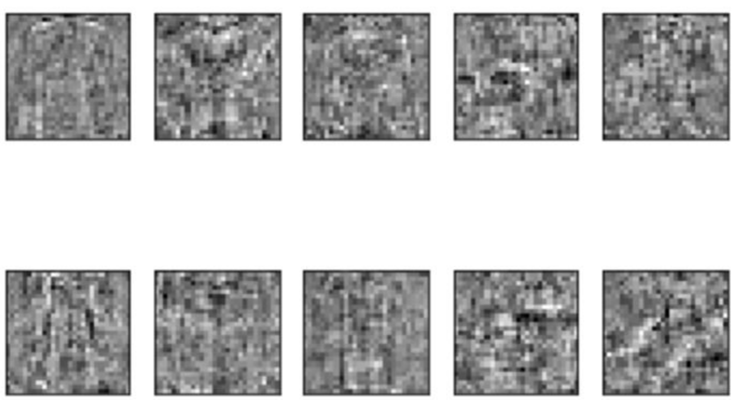

Figure 6.

Then I increase the number of hidden layers to 100. The training runs for longer time and result in 88% for test set score.

3.3. CNN

Finally, I train the dataset with a CNN model with package Keras, which is supposed to provide the best classification. The coefficients of different layers in the CNN model is shown below in Figure 7. The activation function used is ReLU. However, the first training of 50 epochs results in only a score of 56%. The test loss does not improve and remains high during the training. This result does not seem reasonable given that the data is in a good format and the accuracy of CNN is usually high for images. Therefore, I try normalizing the data, dividing all values by 255 and thus transforming them into values between 0 and 1. Although the documentation of Keras does not indicate that such a normalization is required, the training result proves to be much better now.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_7 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
dropout_4 (Dropout)          (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
dropout_5 (Dropout)          (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 3, 3, 128)         73856
_________________________________________________________________
dropout_6 (Dropout)          (None, 3, 3, 128)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               147584
_________________________________________________________________
dropout_7 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 241,546
Trainable params: 241,546
Non-trainable params: 0
_________________________________________________________________
```

Figure 7.

Figure 8. shows the training result of CNN on these 10 classes. The test score is 93%, highest among the three models. To understand the learning process better, I also plot some of the misclassified images with their predictions and correct classes (Figure 9.). It is hard to tell the differences between pullovers, tops, and coats, as expected and told by the t-SNE visualization.

	precision	recall	f1-score	support
T-shirt/top	0.90	0.85	0.87	1000
Trouser	0.99	0.99	0.99	1000
Pullover	0.90	0.90	0.90	1000
Dress	0.92	0.95	0.93	1000
Coat	0.89	0.92	0.90	1000
Sandal	1.00	0.98	0.99	1000
Shirt	0.80	0.80	0.80	1000
Sneaker	0.96	0.97	0.96	1000
Bag	0.99	0.98	0.99	1000
Ankle boot	0.97	0.97	0.97	1000
avg / total	0.93	0.93	0.93	10000

Figure 8.

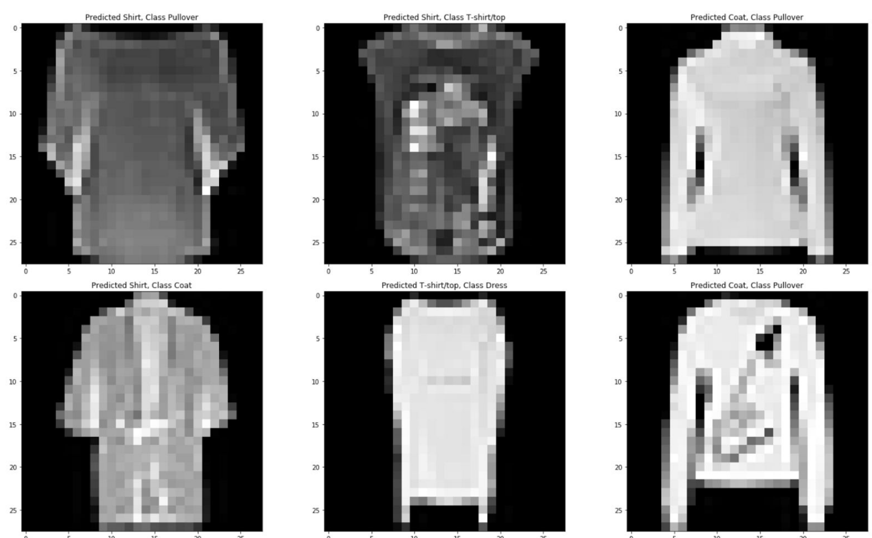

Figure 9.

I further visualize the channels of the image (Figure 10.) after three different activation layers for one specific clothing item. They look very interesting. The channels seem to be learning different features, including the bottom edge, the top left and right corners (shoulders) and the sleeves.

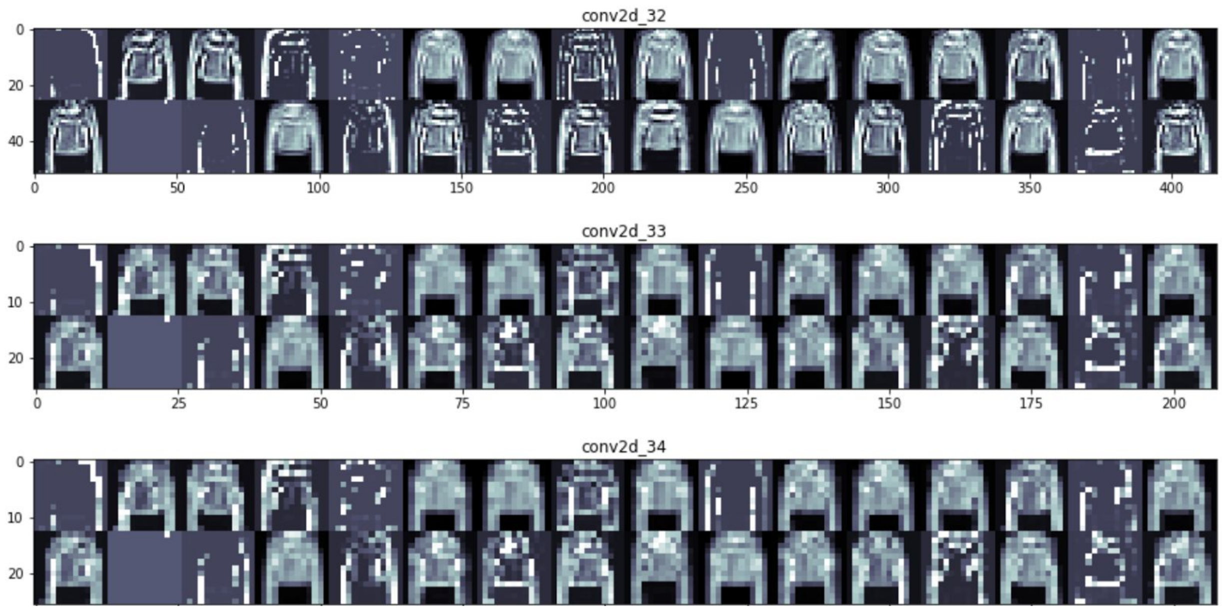

Figure 10.

4. CONCLUSION AND FUTURE WORK

It is clear that the learning algorithms do not work on Fashion MNIST dataset as well as for they work on the original MNIST dataset. Among the three chosen models, CNN gives the best classification result and MLP performs better than logistic regression. It is reasonable given that CNN is considered one of the most powerful image classification method today. The misclassified article are mostly pullovers, coats and tops. Since they do look similar to each other, especially when in grayscale, a better way to classify them may be manually choosing some features after classifying with CNN. However, it could also be very hard since their shapes are almost the same. Some possible features may be the length of sleeves and the pattern and shape of collars.

There still much to do with this dataset. I tend to agree that it would be a good replacement for the original MNIST dataset. Fashion MNIST is easy to implement and is more complex than the handwritten digits. Since the algorithms are getting more and more complicated today, the algorithms that work perfectly on the MNIST dataset might not be good enough. We need a benchmark with higher complexity, which Fashion MNIST could provide. If given more time, I would love to try other learning algorithms as well and compare the result with the original MNIST. In addition, I find it very helpful to visualize the learning process so as to understand the algorithm better, so I would also love to visualize more features for CNN after activation and convolutional layers.

5. REFERENCE

See the links in notebook