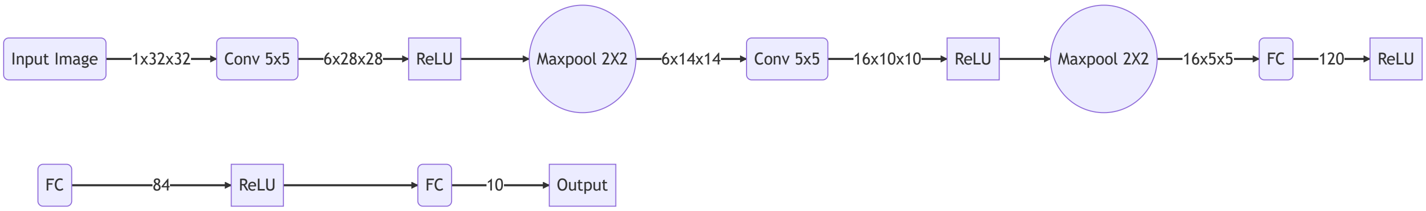
Fashion-MNIST clothing classification

WuYuhang 520030910187

1.LeNet

Model Structure



LeNet consists of a 7-layer structure, including 3 convolutional layers, 2 pooling layers, and 2 fully connected layers.

Since the convolution kernel of the third convolutional layer has the same size as the feature map, it is equivalent to a full connection. So in the final implementation, there are 2 convolutional layers and 3 fully connected layers.

Conv1:

LeNet inputs a picture with a size of 32 × 32 into the convolutional layer, and uses 6 convolution kernels of 5X5 size for convolution to obtain 6 feature map outputs of 28X28 size.

Each convolution kernel is a 5 × 5 matrix, and each number in the matrix is obtained through training; in addition, a bias (bias) is added after the actual convolution operation, and the six convolution kernels need to train (5 × 5 + 1) × 6 = 156 parameters in total.

Pool1:

The pooling layer adopts the maximum pooling layer, the sampling area is 2x2 in size, and the maximum value is taken for each 2 × 2 area, and the output size after the pooling layer is 6x14x14. The pooling layer reduces the feature vector output by the convolutional layer and improves the phenomenon of overfitting.

Conv2:

The convolutional layer 2 continues to use a 5X5 convolution kernel to perform feature extraction on the input 6 feature maps of size 14X14, and finally obtains 16 feature maps of size 10X10.

This layer accepts a convolution kernel with 6 input channels and 16 output channels, and the size of the convolution kernel is 5x5, which requires training (6 × 16 × 5 × 5) + 16 = 2416 parameters.

Pool2:

Exactly the same as pool1, using 2X2 size to perform maximum pooling on the input feature map. The input is 16 feature maps of 10X10 size, and the output size is reduced to half of the original size to 16 feature maps of 5X5 size.

FC1:

Since the convolution kernel of the third layer of convolutional layer is convoluted with the same size as the feature map, it is equivalent to a fully connected layer.

The input is a 16X5X5 feature map, and the output of the fully connected layer is a 120-dimensional vector and is activated by the ReLU function.

FC2:

The input is a 120-dimensional vector, and the output of the fully connected layer is a 84-dimensional vector and is activated by the ReLU function.

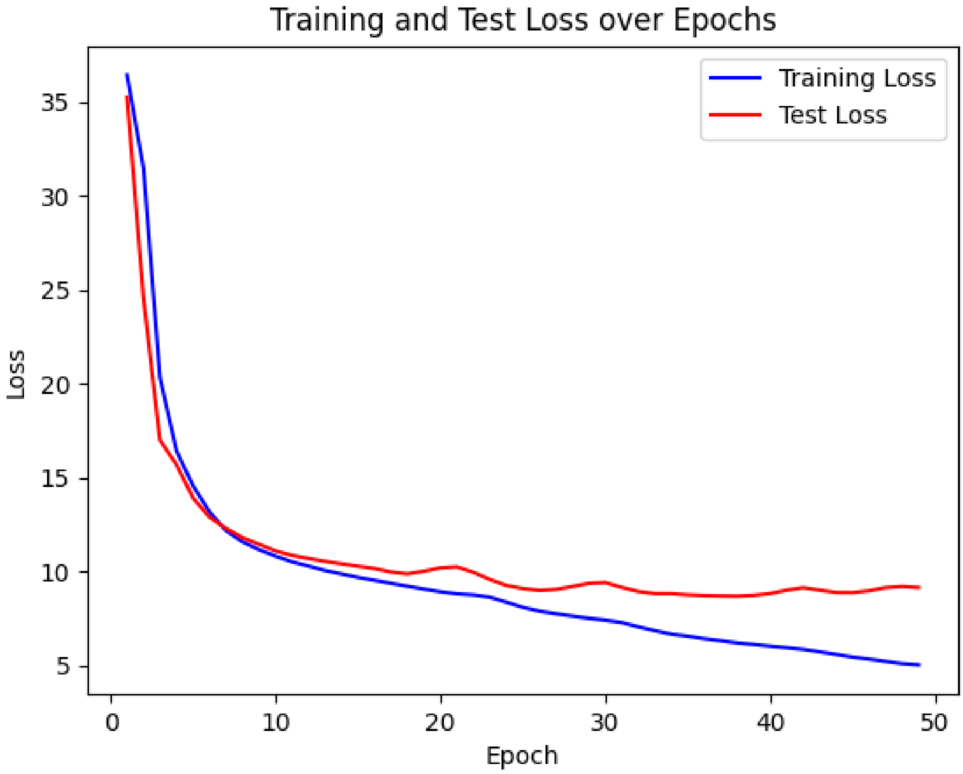
FC3:

The input is a 84-dimensional vector, and the output of the fully connected layer is a 10-dimensional vector and the results correspond to the numbers 0 to 9.

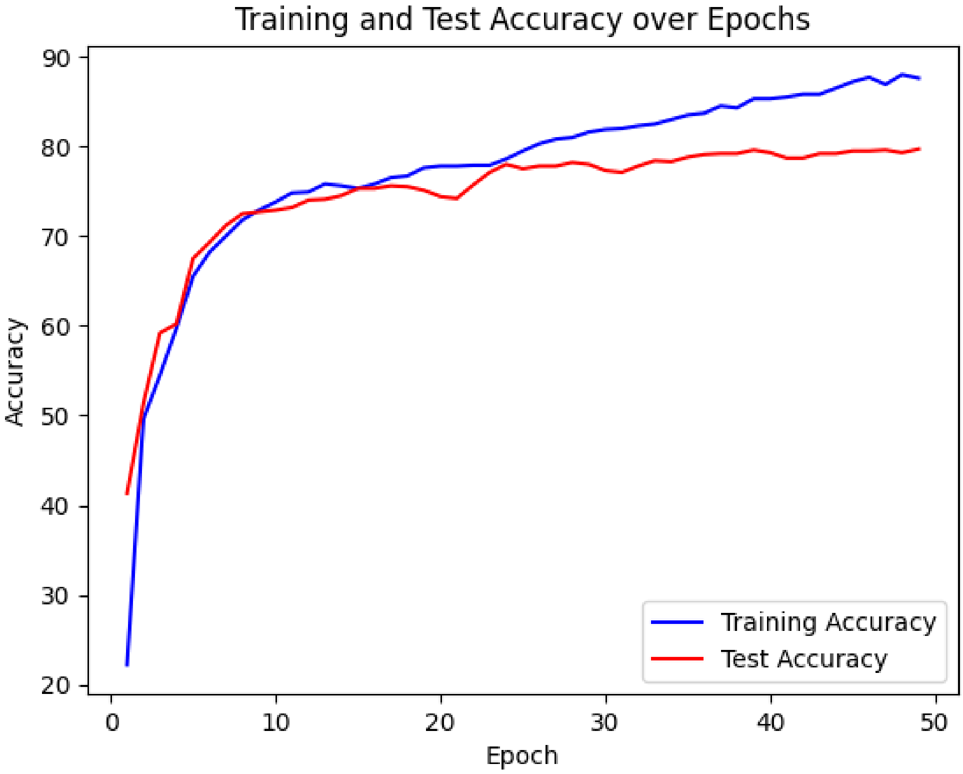
Training process

When training the network, cross-entropy loss is used as the loss function, and Adam is used as the algorithm optimizer.

The loss curve during training is shown in the figure below:



The accuracy curve during training is shown in the figure below:



The model is trained for 50 epochs to get a better fitting effect. The highest accuracy is 80% on the test set, which may be related to the complexity of the model. The complexity of LeNet is too simple. On this basis, increasing the complexity of the network structure may produce better results.

Feature Visualization

1. PCA:

By calculating the covariance matrix of the data matrix, and then obtaining the eigenvalue eigenvector of the covariance matrix, a matrix composed of eigenvectors corresponding to the k features with the largest eigenvalue (that is, the largest variance) is selected. So as to realize the dimensionality reduction of data features.

Step 1:Decentralize the matrix to get a new matrix picture.

$$\overline{X} = \frac{1}{m}\sum_{j=1}^{m} X_{ij}(i = 1, 2, ..., n)$$

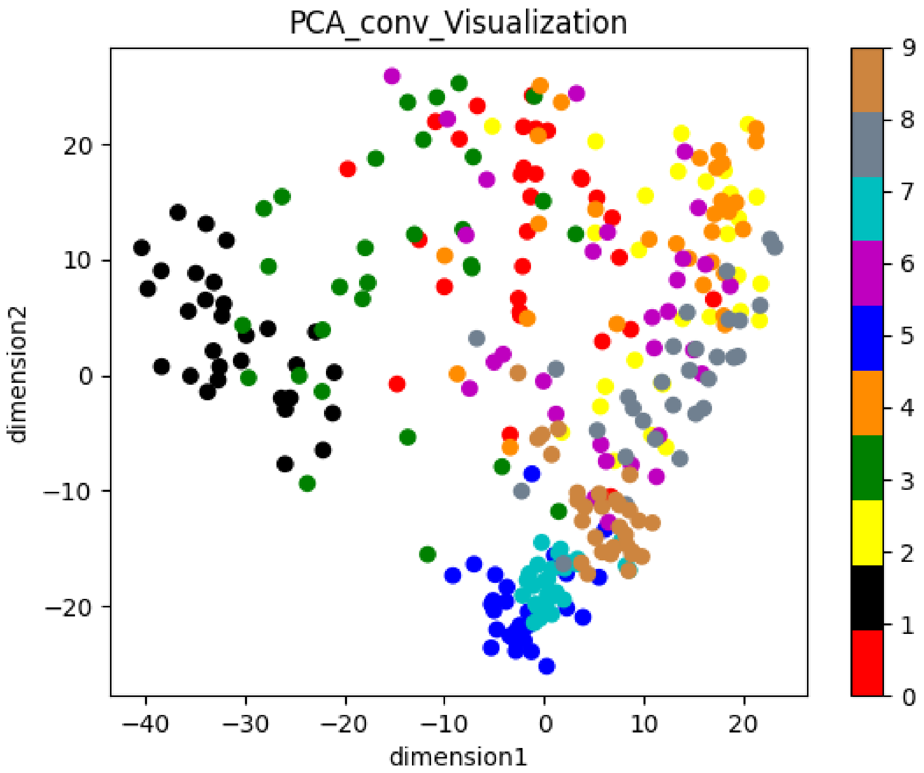
Step 2: Calculate the covariance moment of the decentralized matrix :

$$C = \frac{1}{m-1}X^T X$$

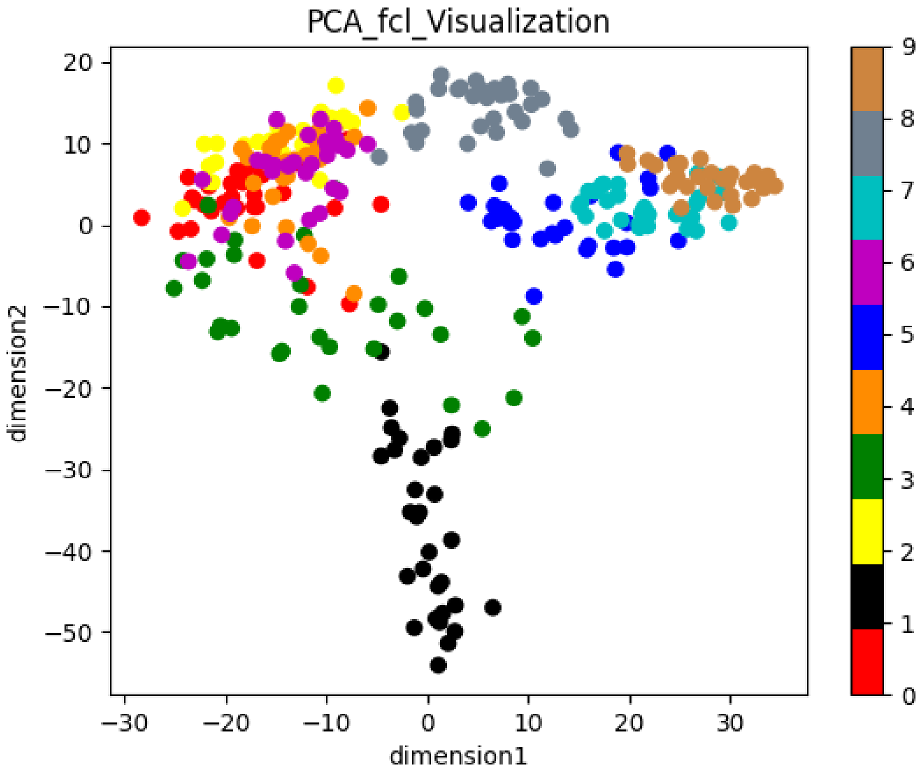
Step 3: Decompose the eigenvalues of the covariance matrix C to find the eigenvalues and eigenvectors of the covariance matrix.

Step 4: Arrange the eigenvectors into a matrix in descending order from left to right according to the corresponding eigenvalues, and take the first k columns to form a matrix. Calculate the sample features after dimensionality reduction to k dimension.

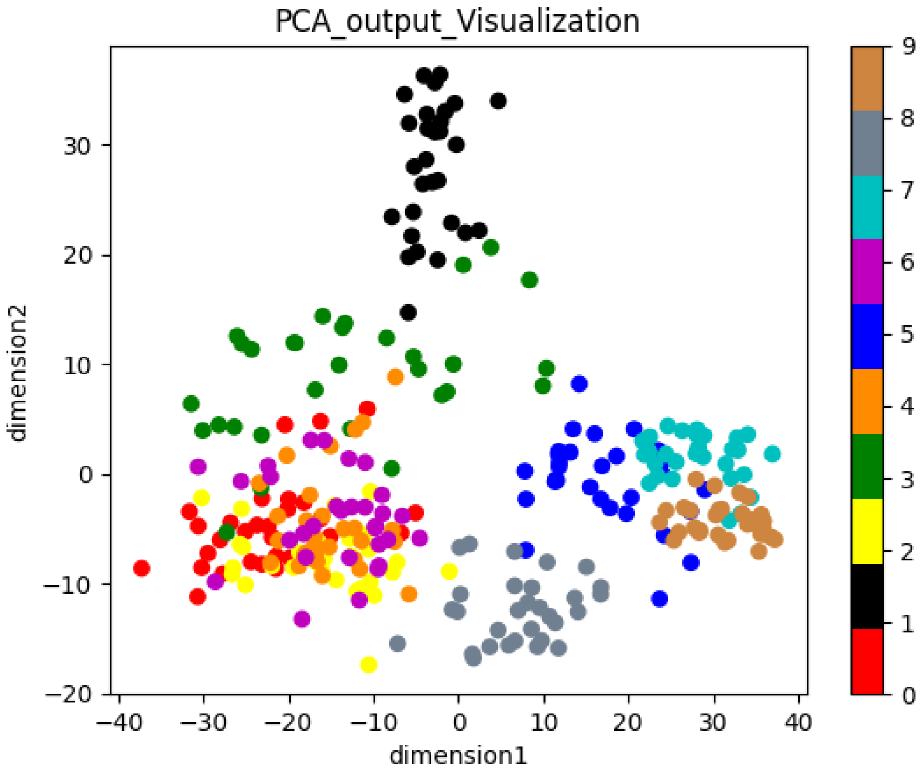
The output features of the second convolutional layer:



Output features of the second fully connected layer:



The features of the output of the last layer result:



2. t-SNE:

The idea of the algorithm is to define a function on the dataset before and after dimensionality reduction to calculate the distance between each point pair. The main method maps data points to probability distributions through affine transformation, and constructs the probability distributions of these points in a low-dimensional space, so that the two probability distributions are as similar as possible.

An alternative idea to optimize the KL divergence of pij and qij is to use the joint probability distribution to replace the conditional probability distribution:

$$C = KL(P||Q) = \sum_i \sum_j p_{i,j} \log \frac{p_{ij}}{q_{ij}}$$

t-SNE assumes that for any i,j:

$$p_{ij} = p_{ji}, q_{ij} = q_{ji}$$

$$p_{ij} = \frac{\exp(- \parallel x_i - x_j \parallel^2 / 2\sigma^2)}{\sum_{k \neq l} \exp(- \parallel x_k - x_l \parallel^2 / 2\sigma^2)}$$

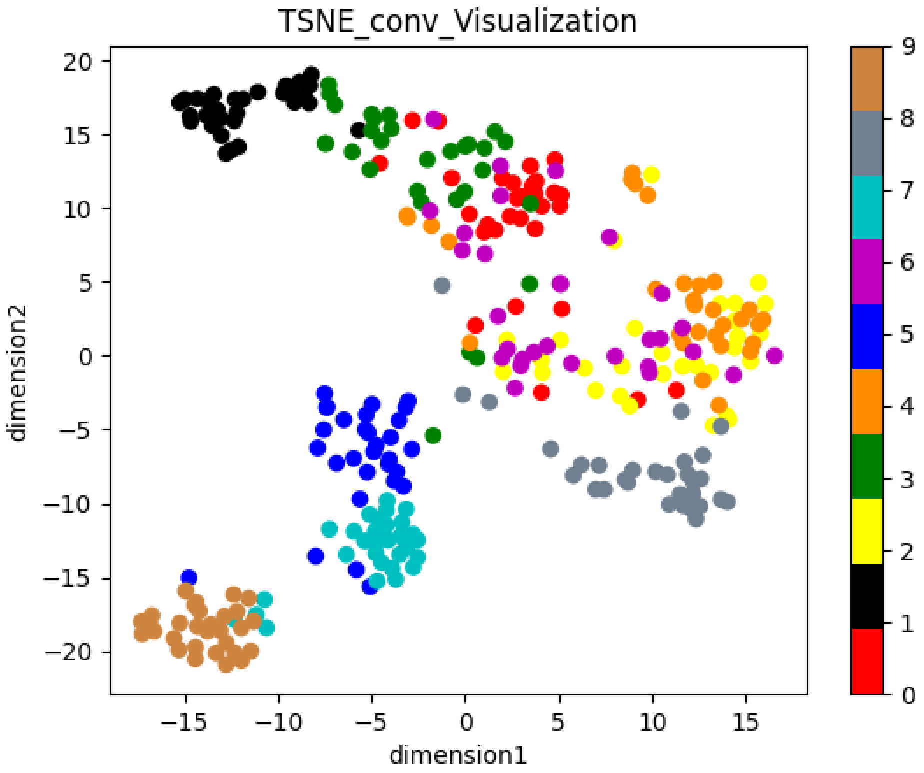
$$q_{ij} = \frac{\exp(- \parallel y_i - y_j \parallel^2)}{\sum_{k \neq l} \exp(- \parallel y_k - y_l \parallel^2)}$$

For the improvement of outliers, the t-SNE approach is to modify the definition of the joint probability distribution as:

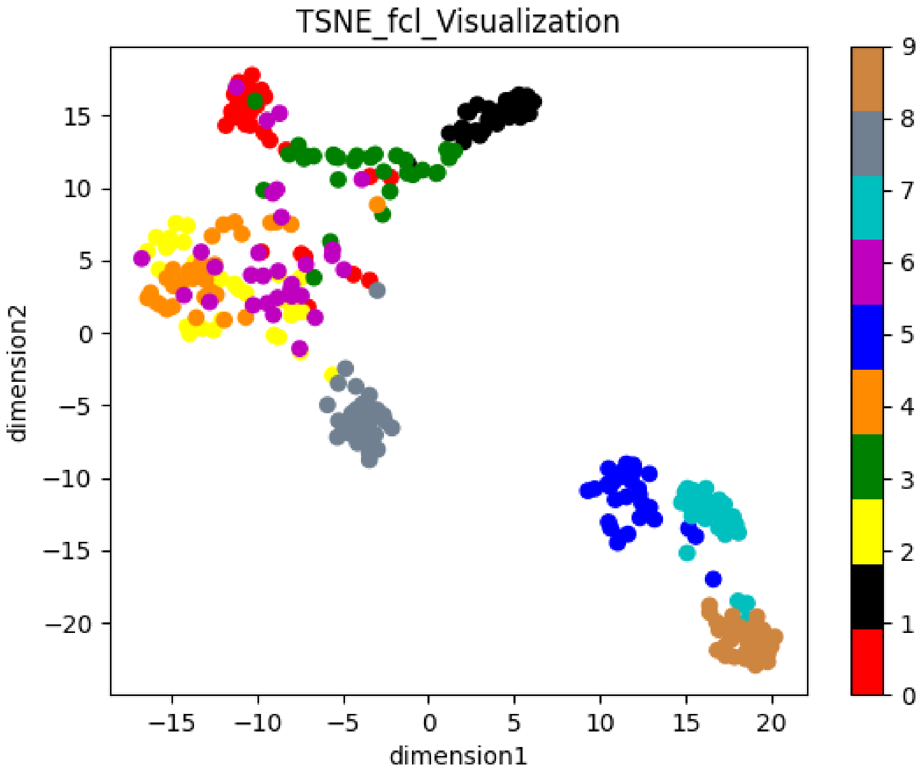
$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2}$$

PCA will produce a little crowded phenomenon in the realization of the results. In order to solve this problem, when t-SNE takes the closest similarity as the optimization goal, the algorithm does not want to divide the point pairs that are similar to each other after dimensionality reduction. For the less similar ones, a larger distance needs to be generated.

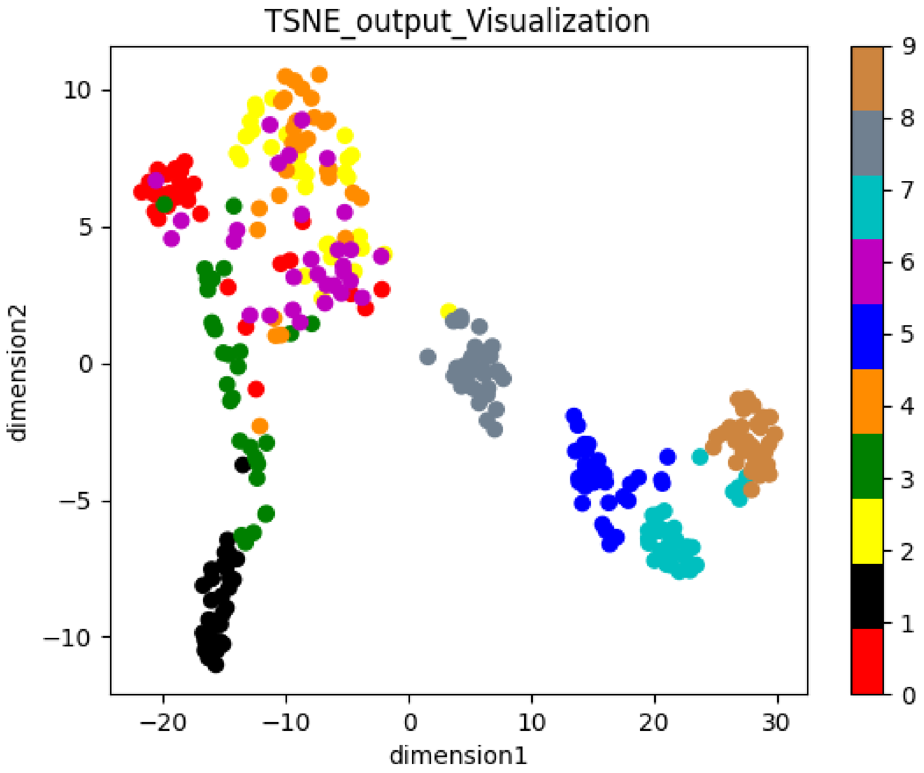
The output features of the second convolutional layer:



Output features of the second fully connected layer:



The features of the output of the last layer result:



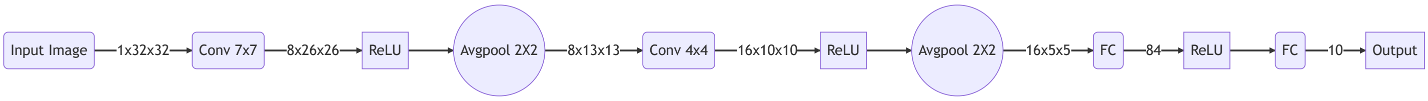
Discussion

1. First of all, comparing the results of PCA and t-SNE, it can be seen that although PCA reduces the data dimension to 2 dimensions, the overlapping data distribution is rather messy. t-SNE is very good at preserving the local structure in high-dimensional data. In the low-dimensional space after dimension reduction, similar data points will gather together to form obvious clusters.

2. It can be seen from the visualization results that the convolutional layer has extracted the local features of the image, which contains the spatial distribution information of different features of the input data. At this time different categories have a preliminary classification after dimensionality reduction.
3. The fully connected layer maps high-dimensional features to classification labels to realize the conversion of features to classification. From the results, it can be seen that each category in the visualization results of the fully connected layer has obvious aggregation.
4. The output of the last layer of the model visualizes the classification confidence of the network for different categories.
5. It can be seen from the results that for half of the data feature learning and classification results are clear, but the other half of the data features are clustered together. This may be related to the simplicity and complexity of the model, and the classification effect on several special categories is not clear. Second, it may be that there are no two distinct dimensions after dimensionality reduction to distinguish categories with similar features.

2.Mynet

Model Structure



The network model I designed myself is based on the structure of LeNet, referring to the structure of its two convolutional layers, and I choose the average pooling layer on the pooling layer. All activation functions use the ReLU function.

The purpose of my cubic convolutional layer is to perform feature extraction on the image. The purpose of using the average pooling layer instead of the maximum pooling layer is to keep the characteristics of the image from undergoing large mutations. Maintain the averaging properties of the features. The last few layers are fully connected layers to classify the features.

The default step size of the convolutional layer I use is 1. For the first convolutional layer, I choose a convolution kernel size of 7X7 to facilitate the extraction of global features, and use 8 different convolution kernels to facilitate the extraction of different features to increase feature richness. In the second convolution layer, I choose the convolution kernel to be 4X4 to facilitate the extraction of further local features, and the output dimension is 16 dimensions to extract more detailed features.

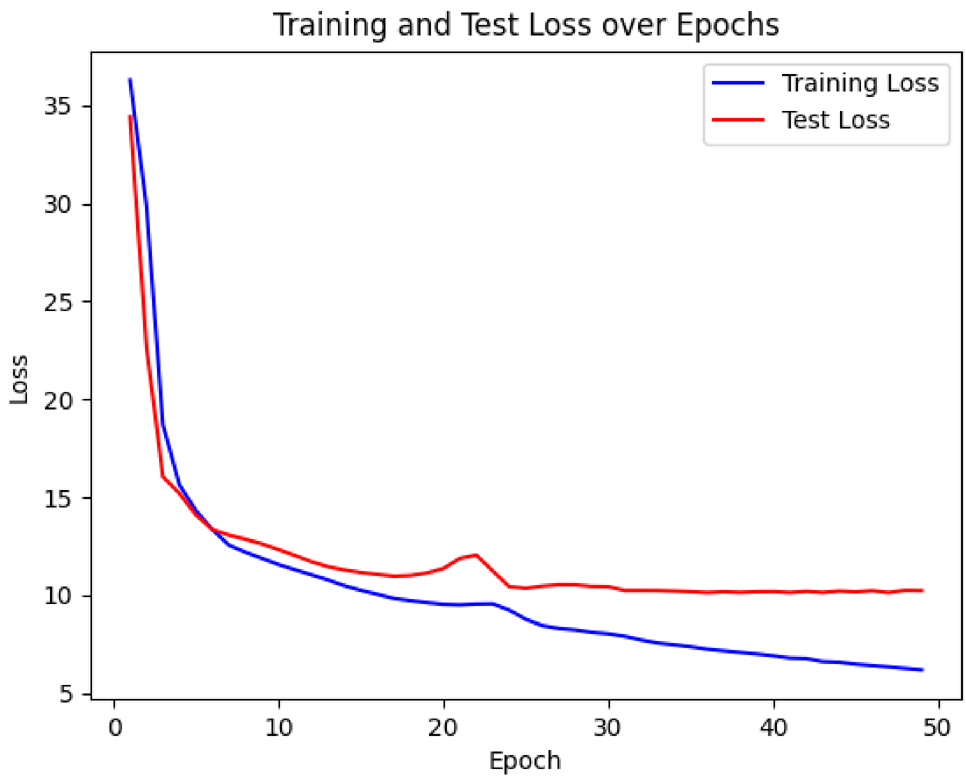
On the pooling layer, the average pooling layer uses a 2X2 size for dimensionality reduction and keeps the average value of the input features from changing significantly under small translations.

On the fully connected layer, I deleted a fully connected layer compared to LeNet and used two fully connected layers. Since the first fully connected layer is equivalent to a 5X5-sized convolutional layer, the second fully connected layer directly classifies the input 84-dimensional vector feature map into 10 dimensions.

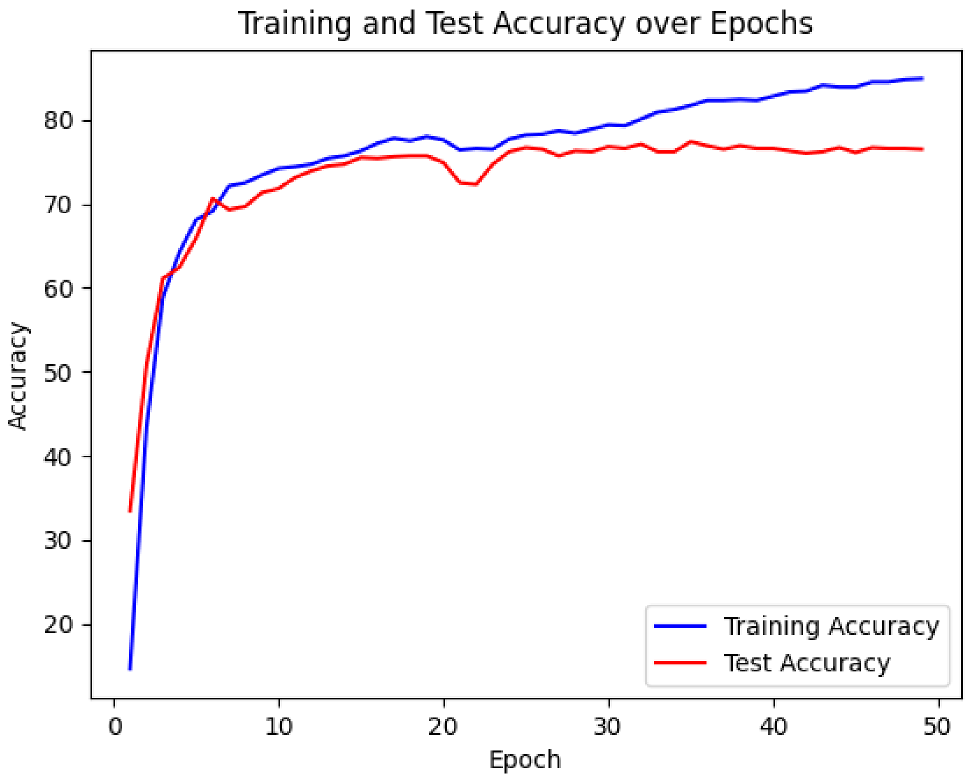
Training process

When training the network, cross-entropy loss is used as the loss function, and Adam is used as the algorithm optimizer.

The loss curve during training is shown in the figure below:



The accuracy curve during training is shown in the figure below:

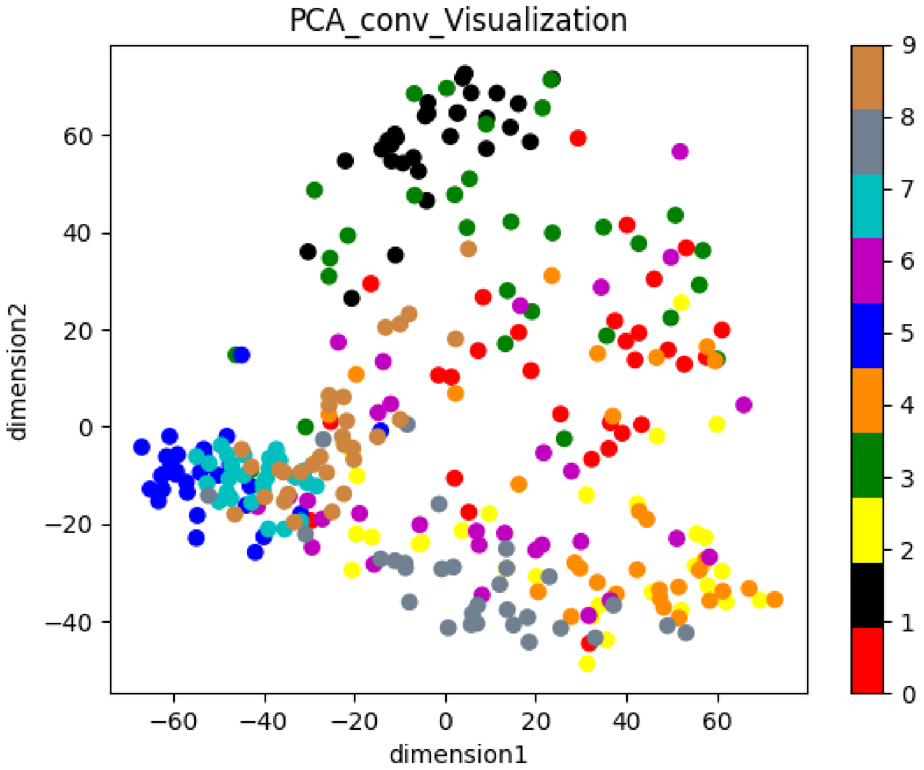


As a result, the model loss curve is similar to LeNet, but the accuracy of the model I designed decreases slightly on the accuracy curve. The overall model was trained for 50 epochs to achieve convergence, and the model accuracy reached 75%-80% on the test set.

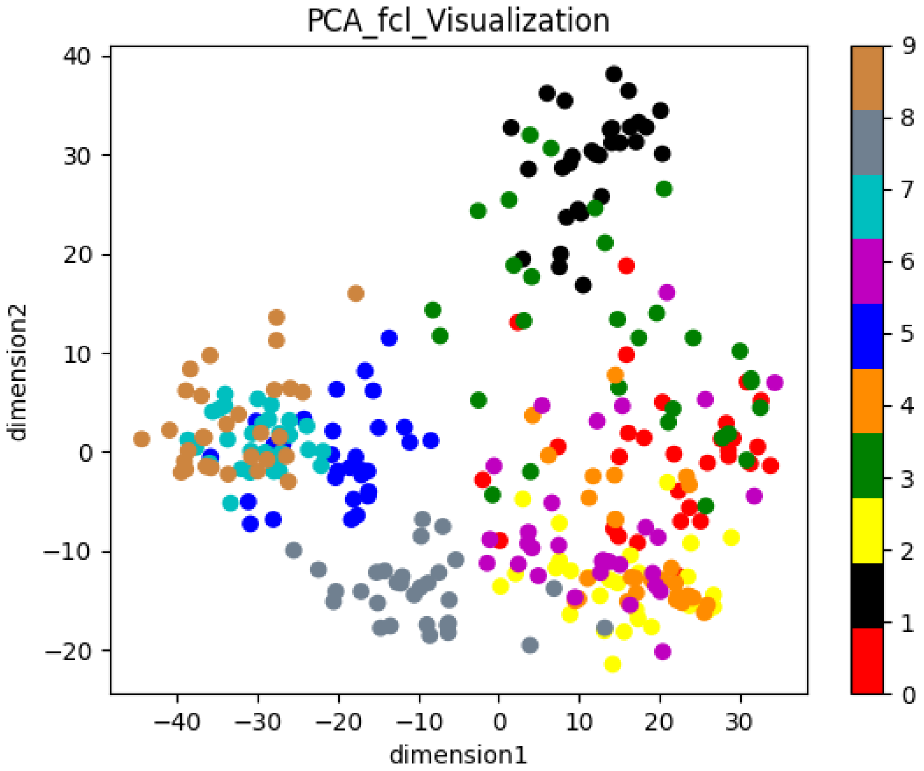
Feature Visualization

1. PCA:

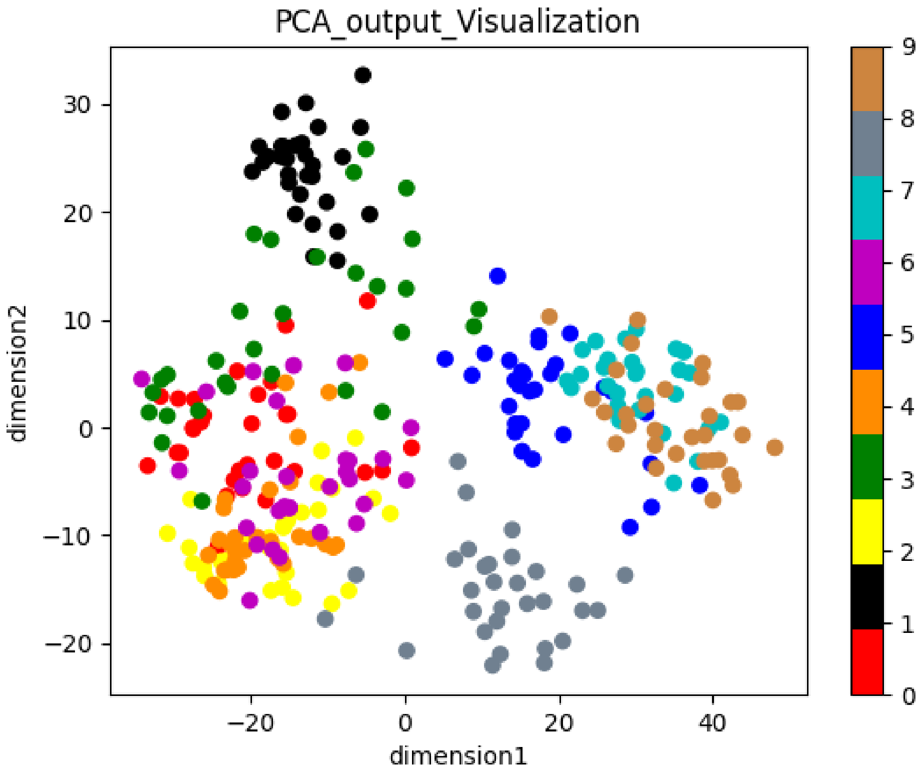
The output features of the second convolutional layer:



Output features of the fully connected layer:

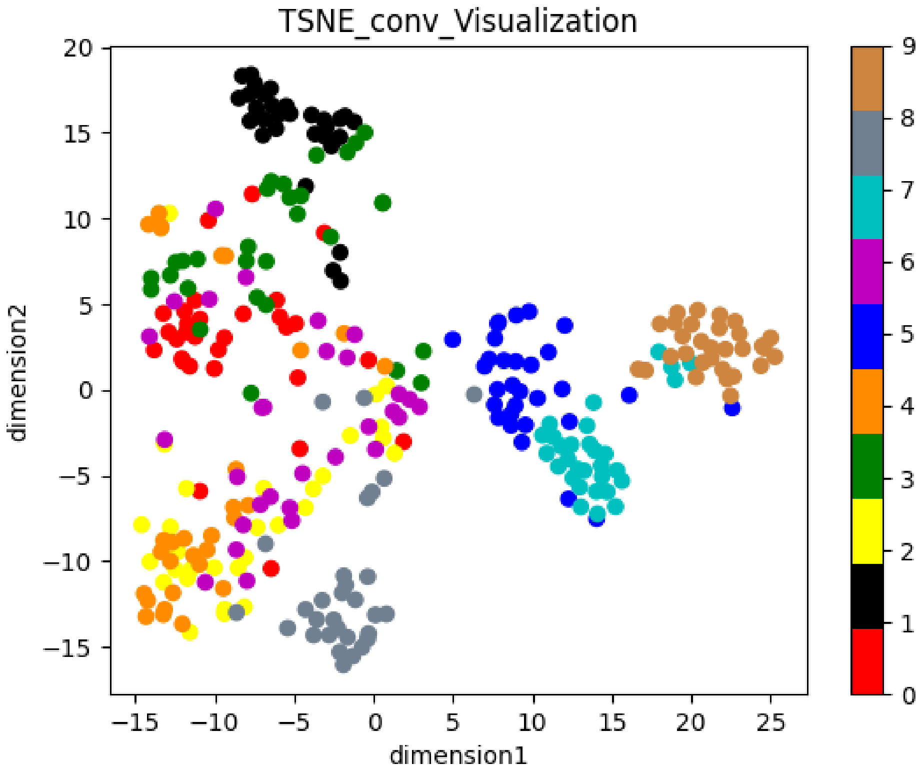


The features of the output of the last layer result:

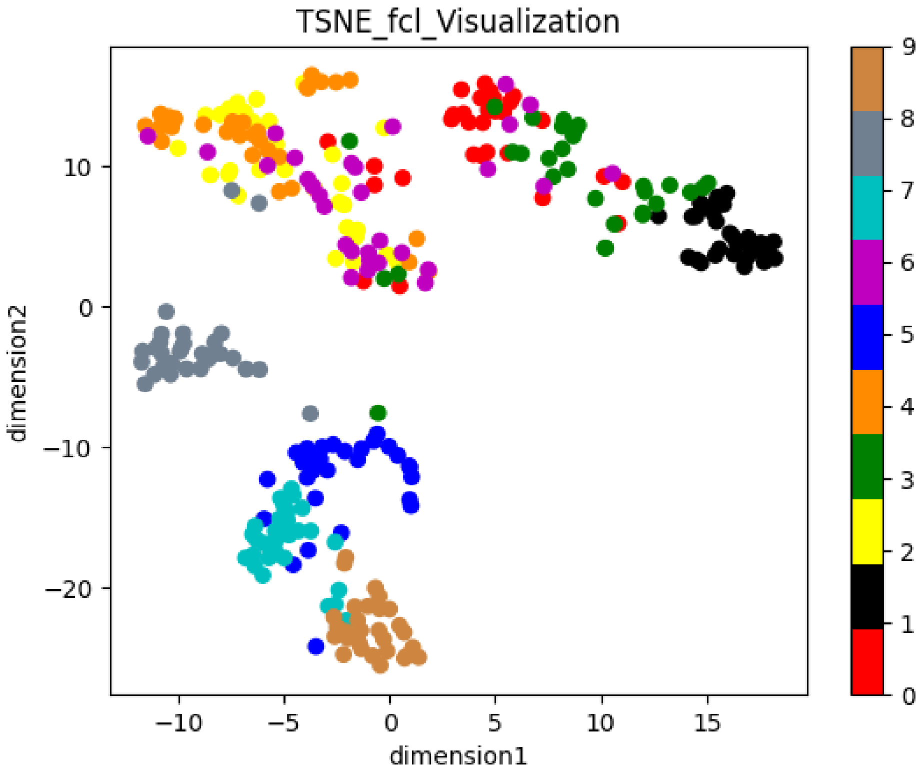


2. t-SNE:

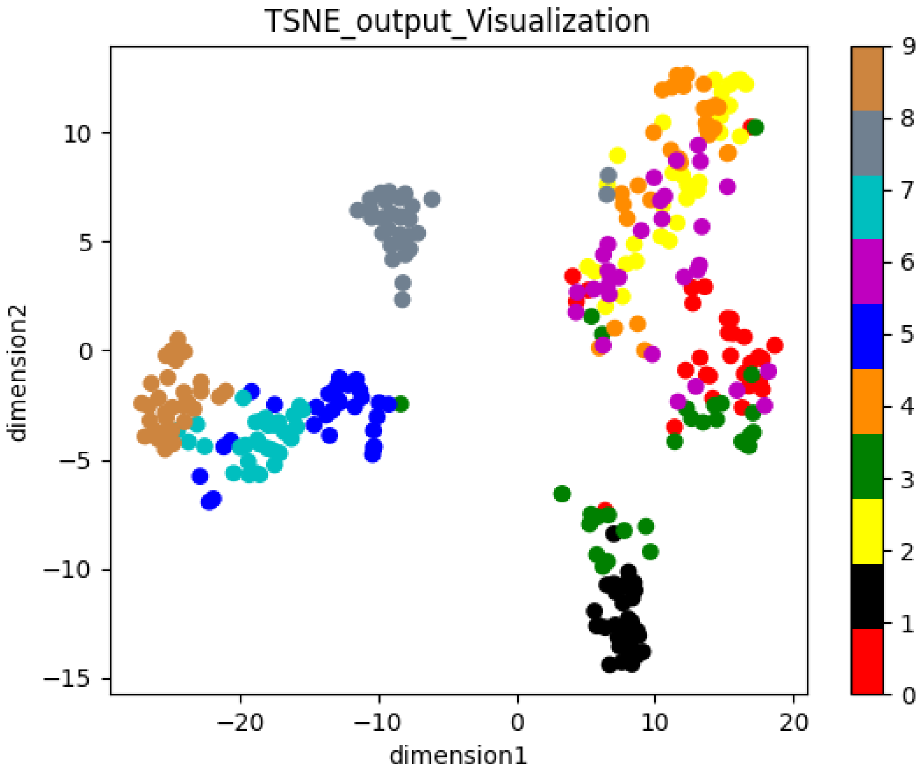
The output features of the second convolutional layer:



Output features of the fully connected layer:



The features of the output of the last layer result:



Discussion

1. Compared with the LeNet network structure, my model is less effective in extracting the category features of green and red nodes. It can be clearly seen from the fully connected layer that the category clustering effect of the green nodes is poor compared to the LeNet network. It may be that the convolution kernel I used is too large and the detailed features are not extracted better.

2. From the results, the visualization effect of the model feature map I designed is similar to that of the LeNet model. It has obvious clustering effect on categories with obvious characteristics, but it is difficult to obtain a good distinguishing effect after dimensionality reduction for categories with similar characteristics.
3. It can be seen from the visualization results that the convolutional layer extracts obvious features from the image. At this point the features have not yet been classified. The fully connected layer maps the features to the classification, and the feature points start to gather. The last layer of feature points completes the aggregation class.

3.Summary

This assignment implements LeNet and the network designed by myself to classify the simplified fashion-MNIST dataset. The classification accuracy of the final training model is about 80%. By visualizing the layer features in the model, it can be seen that the model has limitations in classifying individual categories. And better understand the problem of PCA and t-SNE on the aggregation of data points. Visualization can help to find the design problems of the model and the role of each layer.