Fashion Classification

Yijing Li, Zixin Ma, Runze Xu, Zicong Zhang, Marjan Emadi

Abstract

Classification is one of the main problems in machine learning. In this project, a fashion MNIST dataset, which includes figures of ten different fashion classes, is classified through various machine learning methods using TensorFlow and scikit-learn.
The models of CNN (Convolutional Neural Network) and two different types of architectures (VGG19 and ResNet) are explained in detail, and the results are compared.

Motivation

Nowadays, there are numerous images of clothing and outfits online that could suggest potential business opportunities. For example, by categorizing existing clothing and outfit images from customers in different age groups, we can predict people's preferences and recommend them to targeting customers.
Therefore, a quick and accurate method to search through images and classify them into different categories can be extremely important.

Data

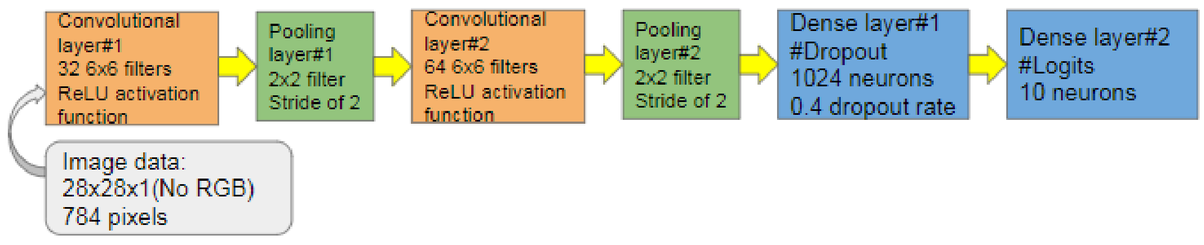
- The dataset contains 70,000 images in total (60,000 training, 10,000 testing).
- Each of the images is a 28*28 greyscale images.
- The images are associated with 10 different classes, each showing a specific type of clothing, such as T-shirt/top, Trouser, Dress, etc..

index	0	1	2	3	4	5	6	7	8	9
Type	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot

Methods

1. CNN (Convolutional Neural Network)

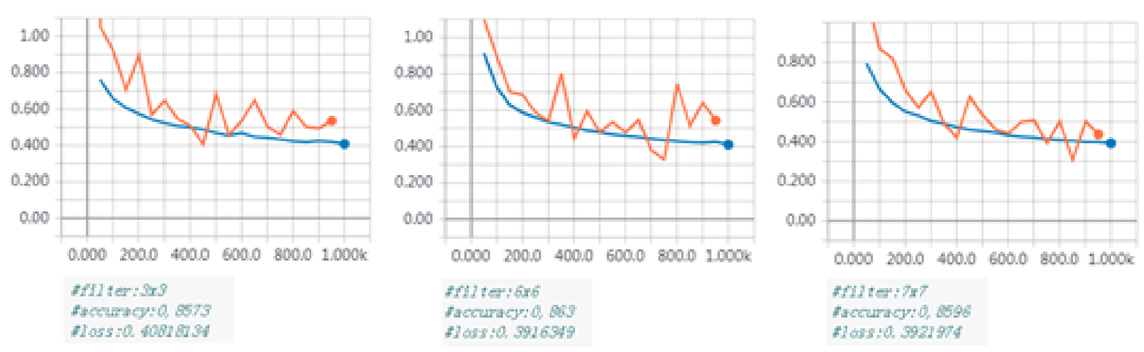
We built a CNN consists of two convolutional layers (extract useful subregions), two pooling layers (downsample the extracted data) and two dense layers (classification based on preceding layers) in the following orders:



The performance of our model is determined by several key factors:

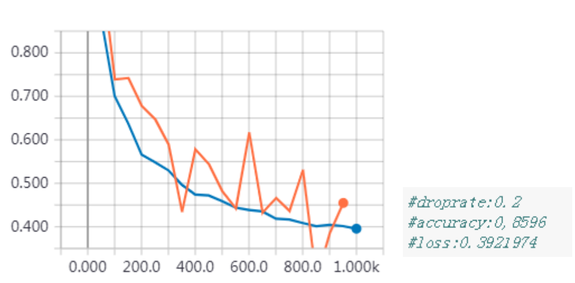
- **Convolutional Layers**

By increasing the size of the filter, we extracted more subregions (features). However, as we allow padding, bigger filter also takes in more empty data section when it approaches to the end of each row and decreases the accuracy.
By comparing performance under different filter size, we found that a 6x6 filter gives the best accuracy:



- **Dense Layer #1:Dropout Layer**

The dropout layer drops part of activation to avoid overfitting:
After the training process, the model becomes too used to predicting based on training data that it actually lowered the performance for test data prediction. As a result, dropping out activations can, to some extent, increase the model's performance when predicting on test data.
We also found that either too low (overfitting) or too high (unable to predict at all) of dropout rate will lower the model performance, and 0.2 will promise the best performance after our experiments. (the performance difference between 0.2 and other dropping rates is limited, but 0.2 is still the best choice)



- **Training Steps:**

We set the training steps to be high enough for maximum performance and also require decent training time for efficiency. In our experiments, the model reaches its maximum performance with 10000 steps.

2. ResNet

We built Resnet-18 model to our task. Resnet has a unique residual block structure. The model is composed by, instead of layers, several residual blocks. This model has been proved effective in classification task. And its residual block makes the model easy to train. Besides trying to achieve best accuracy, we performed experiments on the effect of optimizers, model size, and shortcut mode architecture.

- **Optimizer**

As shown in the figure, Adagrad decreases the loss more effectively than

- **Model Size**

Having too many parameters can cause overfitting and inefficiency in decreasing the loss. Thus, we measured the effect of model size by decreasing the number of parameters in the original resnet-18. As shown in the table, our shrinking strategy increased the model performance.

	original size	1/2 parameters	1/4 parameters	1/8 parameters	1/16 parameters
loss/ accuracy	0.62/83.5%	0.43/87.4%	0.37/87.3%	0.38/87.3%	0.40/86.8%

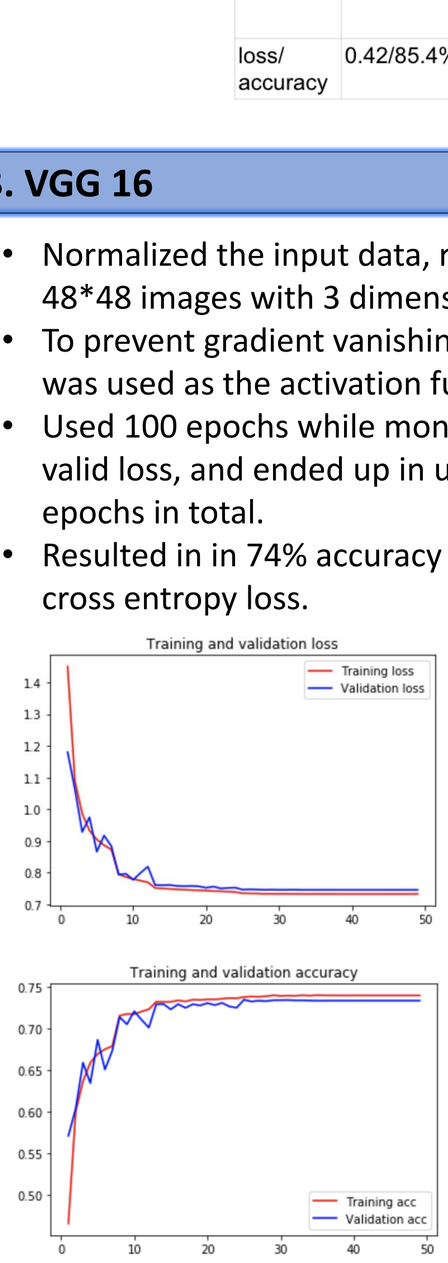
- **Shortcut of Res-block**

We also measured the influence of Resnet's shortcut. The effect of shortcuts has been explored thoroughly. It can help propagate gradient to shallow layers of the model. Because recent models are usually very deep, the shortcut is indispensable to ensure that these models work.

	All Shortcuts	only 4th shortcut	only 3rd shortcut	only 2nd shortcut	only 1st shortcut
loss/ accuracy	0.42/85.4%	0.47/84.3%	0.51/83.1%	0.49/83.8%	0.63/77.8%

3. VGG 16

- Normalized the input data, resized to 48*48 images with 3 dimensions.
- To prevent gradient vanishing, ReLu was used as the activation function.
- Used 100 epochs while monitoring the valid loss, and ended up in using 49 epochs in total.
- Resulted in in 74% accuracy and 0.73 cross entropy loss.



Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 48, 48, 3)	0
block1_conv1 (Conv2D)	(None, 48, 48, 64)	1792
block1_conv2 (Conv2D)	(None, 48, 48, 64)	36928
block1_pool (MaxPooling2D)	(None, 24, 24, 64)	0
block2_conv1 (Conv2D)	(None, 24, 24, 128)	73856
block2_conv2 (Conv2D)	(None, 24, 24, 128)	147584
block2_pool (MaxPooling2D)	(None, 12, 12, 128)	0
block3_conv1 (Conv2D)	(None, 12, 12, 256)	295168
block3_conv2 (Conv2D)	(None, 12, 12, 256)	590080
block3_conv3 (Conv2D)	(None, 12, 12, 256)	590080
block3_pool (MaxPooling2D)	(None, 6, 6, 256)	0
block4_conv1 (Conv2D)	(None, 6, 6, 512)	1180160
block4_conv2 (Conv2D)	(None, 6, 6, 512)	2359808
block4_conv3 (Conv2D)	(None, 6, 6, 512)	2359808
block4_pool (MaxPooling2D)	(None, 3, 3, 512)	0
block5_conv1 (Conv2D)	(None, 3, 3, 512)	2359808
block5_conv2 (Conv2D)	(None, 3, 3, 512)	2359808
block5_conv3 (Conv2D)	(None, 3, 3, 512)	2359808
block5_pool (MaxPooling2D)	(None, 1, 1, 512)	0

4. Traditional Methods

We used four different traditional ML method for our problem and got the following accuracies:

Methods	KNN	SVM	RandomForest	Adaboost
Accuracy (%)	87	89	86	58

Discussion & Conclusion

From the results of different methods on solving the fashion classification problem, we can conclude that the accuracy of these methods are close, which is around 87%.
However, the running time of VGG, ResNet and CNN is significantly faster than that of the traditional methods such as SVM and boosting in sklearn.

UC San Diego
JACOBS SCHOOL OF ENGINEERING
Electrical and Computer Engineering