Clothing Type Classification

Zhixiang Meng
School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
z9meng@uwaterloo.ca

Abstract

Fashion MNIST dataset [1] is a newly collected MNIST-like dataset which is expected to help quickly check and prototype Machine Learning algorithms as a benchmark dataset. In this project, we apply several classical classification methods, including Perceptron, Ridge Regression, LASSO, Logistic Regression, K-Nearest Neighbours, Support Vector Machine, and Convolutional Neural Network, to this Fashion MNIST dataset to do the clothing type classification and compare their performances. We report the best test accuracy to be 94.67% by using a simple Convolutional Neural Network model. Inspired by the data preparation procedure of Fashion MNIST dataset, we also process another two datasets of fashion products images, the Apparel Classification with Style (ACS) dataset created by Bossard et al. [2] and the Fashionwear (FW) dataset prepared by Subramanya et al. [3], in a similar way, and do the classification by using AlexNet [4] and VGG11 [5] with modified structures. We report the best test accuracy to be 83.24% by using modified VGG11 on FW dataset. By converting sample images into smaller grayscale images, the training time can be largely reduced while similar classification accuracy can still be achieved.

1 Introduction

Image Recognition and Classification is a popular topic nowadays, especially after people proposed many efficient Neural Network structures using convolutional layers. Meanwhile, clothing is a necessary and important part of our daily life, and people may spend much time in searching, selecting and buying clothes. With the booming of online shopping, people are more likely to buy clothes online and thus there is an increasing demand for clothing detection, classification and searching service. So, in our project, we conduct clothing type classification by using three datasets [1] [2] [3] and hope we can find a better solution to this problem.

Fashion MNIST dataset is newly proposed by Xiao et al. in 2017 [1]. It is expected to replace MNIST dataset created by LeCun et al. [6] as a benchmark dataset for Machine Learning algorithms but be more challenging at the same time. As images in Fashion MNIST dataset are in small size, the same as MNIST dataset, but with more complicated data, it can serve a good but more challenging benchmark dataset for checking and prototyping Machine Learning algorithms. So, we apply different Machine Learning algorithms to Fashion MNIST dataset, solve the clothing type classification problem and compare their performances.

In [2], Bossard et al. created the ACS dataset and did apparel classification based on it. They used one-vs-all linear Support Vector Machine, Random Forest, and Transfer Forest to do the classification, achieving 35.03%, 38.29% and 41.36% accuracy on average respectively. Based on their work, Lao and Jagadeesh [7] used a modified AlexNet model with pre-trained weights by ImageNet, and after fine tuning, achieved 50.2% average accuracy for clothing type classification of the

Submitted to Introduction to Machine Learning (CS698, 2017 Fall). Do not distribute.

ACS dataset. In [3], Subramanya et al. created another dataset, the FW dataset, for clothing type classification. They used AlexNet to do the classification and achieved 87.5% average accuracy.

Inspired by the preparation procedure and the classification results of Fashion MNIST [1], we think that the most important aspect of classifying clothing types is to recognize the outline of fashion products. After all, those 28 x 28 grayscale images in Fashion MNIST cannot preserve many details of clothes. So, if we can classify clothing types by using much smaller images, then the training and test time will be much shorter, which is quite desirable. Therefore, in our project, we process the ACS dataset and the FW dataset as Xiao et al. did for Fashion MNIST [1], and use the processed datasets to train two VGG11 models and two AlexNet models with modified structures for clothing type classification.

Another concern here is that both [7] and [3] did the classification by using AlexNet in slightly different ways but achieved quite different accuracy on average. We also want to figure out the possible reasons behind that.

2 Technical Approach

Classification of Fashion MNIST Dataset

In [1], Xiao et al. explained in detail how this Fashion MNIST dataset was created and performed several tests on it by using different classification algorithms. But they did not use Neural Network method to do the test. So, in our project, we first uses the algorithms we learned in this course to do the classification, then build a Convolutional Neural Network model and see whether it will perform better with higher test accuracy. To be specific, we choose to use Perceptron, Ridge Regression, LASSO, Logistic Regression, K-Nearest Neighbours, and Support Vector Machine, as well as Convolutional Neural Network.

Perceptron is a classification algorithm proposed by Rosenblatt in 1957 [8]. It is essentially a single-layer Neural Network, the simplest one, for binary classification. To achieve good performance, i.e. reach some low error rate, it requires the data to be roughly linearly separable; Otherwise, the training process would be hard to converge and the model would probably fail to work. Even though it is for binary classification, it can also be used to solve multi-class classification problems by using one-vs-one or one-vs-all strategy.

$$\mathbf{y} \times sign(\langle \mathbf{w}, \mathbf{x} \rangle) > 0$$

Ridge Regression and LASSO [9] are quite similar linear models. Ridge Regression is to solve least square problems with L2 regularization on weights and LASSO uses L1 regularization instead. As they use different regularizations, Ridge Regression favours small weights but LASSO encourages sparsity of weights.

$$\min_w \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_2^2$$

$$\min_w \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_1$$

Logistic Regression is another linear model for classification, where logistic function is used to model the probabilities of different outcomes. [10] Different from Ridge Regression and LASSO, it is not to solve least square problems but to solve maximum likelihood estimation problems. It can also be implemented as penalized regression by using L1, L2 or other regularization. Similarly, if using L1 regularization, it encourages sparsity, and if using L2 regularization, it favours small weights.

$$\min_w \sum_i \log\left(1 + \exp(-y_i w^T x_i)\right) + \lambda \|\mathbf{w}\|_2^2$$

$$\min_w \sum_i \log\left(1 + \exp(-y_i w^T x_i)\right) + \lambda \|\mathbf{w}\|_1$$

K-Nearest Neighbours algorithm (KNN) is another commonly-used classification method. [11] Different from all the methods above, it is non-parametric, which means there are no weights to be trained. The classification is based on the distances between the test sample and all the training

samples, so it is also called instance-based learning. One important part of KNN is to use Cross-Validation to select the value of K, i.e. the number of neighbours to be considered when classifying a new sample and the outcome will be the majority vote of those K nearest neighbours. Moreover, instead of using naive majority vote, i.e. each neighbour of the K nearest ones has the same weight in the voting, it can be implemented as each neighbour has different weight which is negatively related to the distance between the new sample and that neighbour.

Support Vector Machine (SVM) [12] is again a commonly-used classifier, which is closely related to Perceptron. One big difference is that Perceptron is only to find a hyperplane to separate the data into two classes but SVM is also trying to maximize the margin between the separating hyperplane and two classes of data. As a result, Perceptron can be trained online, i.e. its weights can be updated when training samples come one by one, which is quite hard for SVM.

Convolutional Neural Network (CNN) is a special type of Neural Network and is widely used in Image Recognition and Classification. For example, Yann designed LeNet-5 for handwritten character recognition [13]. CNN uses small filters to do convolution to generate inputs for next layer instead of calculating weighted sum of outputs from nodes in one layer for each node in next layer. Considering the efficiency and effectiveness of CNN in Computer Vision, we try to apply CNN in classifying clothing types in our project. For Fashion MNIST, we use a very simple CNN model as showed in 1.

a simple CNN model
Input (28 x 28 grayscale images)
conv3-32
maxpool
conv3-64
maxpool
conv3-128
maxpool
fc-128
fc-10
softmax

Table 1: Structure of a Simple CNN Model

Classification of ACS and FW Dataset

Original VGG11	Modified Model for ACS	Modified Model for FW
Input (224x224 RGB images)	Input (32x32 grayscale images)	Input (32x32 grayscale images)
conv3-64	conv3-32	conv3-32
maxpool	maxpool	maxpool
conv3-128	conv3-32	conv3-32
maxpool	maxpool	maxpool
conv3-256	conv3-64	conv3-64
conv3-256	conv3-64	conv3-64
maxpool	maxpool	maxpool
conv3-512	conv3-256	conv3-256
conv3-512	conv3-256	conv3-256
maxpool	maxpool	maxpool
conv3-512	conv3-256	conv3-256
conv3-512	conv3-256	conv3-256
maxpool	maxpool	maxpool
fc-4096	fc-256	fc-256
fc-4096	fc-256	fc-128
fc-1000	fc-15	fc-13
softmax	softmax	softmax

Table 2: Structures of Original VGG11 Model and Modified Models

Original AlexNet	Modified Model for ACS	Modified Model for FW
Input (224x224 RGB images)	Input (32x32 grayscale images)	Input (32x32 grayscale images)
conv3-3	conv3-3	conv3-3
conv3-96	conv3-32	conv3-32
maxpool	maxpool	maxpool
conv3-192	conv3-64	conv3-64
maxpool	maxpool	maxpool
conv3-288	conv3-128	conv3-128
conv3-288	conv3-128	conv3-128
conv3-256	conv3-96	conv3-96
maxpool	maxpool	maxpool
fc-4096	fc-256	fc-256
fc-4096	fc-256	fc-128
fc-1000	fc-15	fc-13
softmax	softmax	softmax

Table 3: Structures of Original AlexNet Model and Modified Models

For ACS [2] and FW [3] dataset, we choose to build models by modifying the original structures of VGG11 [5] and AlexNet [4] to do the classification. VGG11 [5] and AlexNet [4] are two famous model structures using CNN to do Image Recognition and Classification. The models consist of convolutional layers, max pooling layers, and fully-connected layers. In our project, as we resize the data samples to the size of 32 x 32 by downsampling and there are 13 or 15 categories instead of 1,000 categories to classify, we modify the structure of VGG11 and AlexNet accordingly for better fitting, as shown in 2 and 3.

3 Evaluation

Classification of Fashion MNIST Dataset

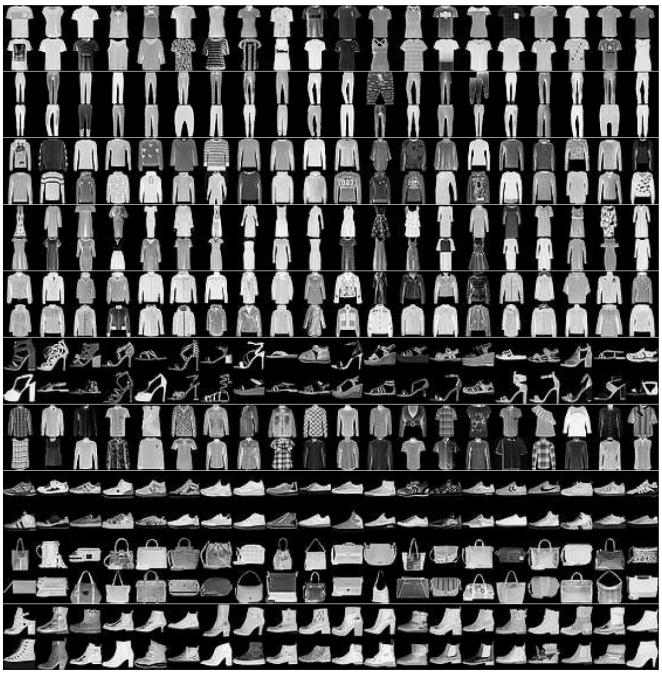
Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

Table 4: Class Names and Example Images in Fashion MNIST Dataset

The dataset we used in this part is Fashion MNIST dataset [1]. It contains 70,000 images, each of which is a 28 x 28 grayscale image of some fashion product, and in total there are 10 categories. There are 60,000 samples in the training dataset, 6,000 samples for each category, and there are 10,000 samples in the test dataset, 1,000 samples for each category. Details can be found in 4.

By using the CNN model 1, we achieved 94.67% test accuracy on average, training over 5 epochs in around 1,000s. The classification results of other Machine Learning algorithms are listed in 5. Obviously, CNN outperformed all the other algorithms as expected. This also shows that even by using small images, clothes can be correctly classified with low error rate.

As we can see, the performance of Perceptron was the worst, with relatively low accuracy but long training time. This is expected, as Perceptron indeed requires the data to be linearly separable, but image datasets, like Fashion MNIST, usually are not linearly separable. We can also find that Ridge Regression was the fastest and its training time was less than 20 seconds. The reason is that there almost always is a closed-form solution for Ridge Regression. For Perceptron, Ridge Regression/LASSO and Logistic Regression, it is easy to conclude that by using L1 penalty, the sparsity of weights was much larger, i.e. there were more zeros among the trained weights, compared with using L2 penalty. But using L1 penalty took longer time for training, than using L2 penalty. This is due to different computational complexity. For L1 penalty, it is equivalent to a quadratic programming problem, while for L2 penalty there is usually an analytical solution. And Logistic Regression, KNN and SVM performed similarly well, achieving relatively high accuracy. But the training time of SVM was clearly longer than that of the others, indicating that it took longer for SVM to converge.

Algorithm	Regularization	Test Accuracy	Training Time	Sparsity
Perceptron	No Penalty	77.97%	331.90s	0%
Perceptron	L1 Penalty	77.32%	908.04s	81.53%
Perceptron	L2 Penalty	74.97%	352.82s	0%
Ridge Regression	L2 Penalty	76.21%	16.66s	0%
LASSO	L1 Penalty	76.18%	31.47s	48.09%
Logistic Regression	L1 Penalty	84.23%	578.47s	37.56%
Logistic Regression	L2 Penalty	84.11%	282.95s	0%

Algorithm	K	Distance Type	Vote Weights	Test Accuracy	Training Time
KNN	5	manhattan	uniform	86.15%	713.92s
KNN	5	euclidean	uniform	85.54%	649.34s
KNN	5	manhattan	distance	86.23%	688.97s
KNN	5	euclidean	distance	85.77%	690.19s

Algorithm	Kernel	Test Accuracy	Training Time
SVM	linear	84.63%	738.35s
SVM	sigmoid	83.13%	1056.82s
SVM	gaussian	84.61%	942.76s

Table 5: Test Accuracies of Classification on Fashion MNIST Dataset

Classification of ACS and FW Datasets

The datasets we used in this part are the ACS dataset [2] and the FW dataset [3]. The ACS dataset contains over 80,000 RGB images of different apparels belonging to 15 categories. The sizes of those images are different and the maximum length of row and column is 320 pixels. It is randomly split into training dataset which contains 71,626 images and test dataset which contains 17,858 images. Details of the ACS dataset can be found in 6. The FW dataset contains 12,759 RGB images of different clothes belonging to 13 classes. The sizes of those images are also different. We randomly chose 10% of it as test data, so there are 11,483 training samples, and there are 1,276 test samples. Details of the FW dataset can be found in 7.

Similarly with the procedure described in [1], we processed these two datasets by resizing, sharpening, extending, negating and converting. First, resize the images by downsampling such that the longest edge is 32 pixels and keep the aspect ratio of each image unchanged. Second, sharpen the

resized images. Third, extend the shorter edge of each image to be 32 pixels by filling white pixels and put the image in the center. Forth, negate the intensities. Last, convert them to grayscale images. After that, we got two datasets containing 32 x 32 grayscale images of fashion products belonging to different clothing classes. Here are some examples shown in 1 and 2.

Class Name	Data Size
Long Dress	12,622
Suit	7,573
Shirt	1,784
Coat	11,338
Undergarment	6,927
T-shirt	1,784
Jacket	11,719
Uniform	4,194
Blouses	1,121
Cloak	9,371
Sweater	6,515
Vest	938
Robe	7,262
Short Dress	5,360
Polo Shirt	976
Total	89,484

Table 6: Details of ACS Dataset

Class Name	Data Size
Blazers	1,793
Dresses	1,013
Jeans	1,313
Jumpsuits	947
Kurtis	1,065
Palazzo trousers	582
Sarees	1,124
Sherwani	402
Shirts	1,218
Suits	390
Tops	1,050
Trousers	1,037
T-shirts	825
Total	12,759

Table 7: Details of FW Dataset

Figure 1: Examples of Processed Images of ACS Dataset

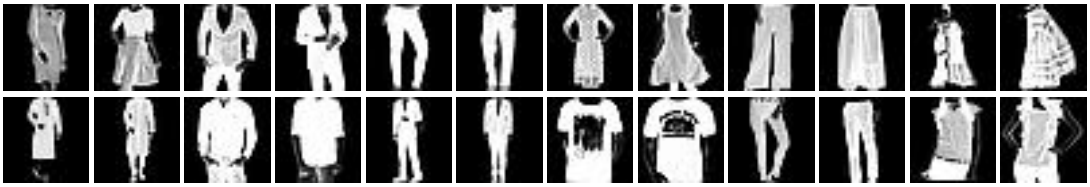


Figure 2: Examples of Processed Images of FW Dataset

Using the processed datasets, we built and trained two VGG11 models and two AlexNet models with modified structures, as stated in 2 and 3, over 50 epochs respectively. The test accuracies on average that we achieved are shown in 8. Clearly, the accuracies for classifying processed FW dataset are much better than those for ACS dataset. For each epoch, it took around 1 min (AlexNet) and 2 mins (VGG11) for FW dataset and around 7 mins (AlexNet) and 14 mins (VGG11) for ACS dataset.

Test Accuracy	Modified VGG11	Modified AlexNet
ACS Dataset	38.86%	38.37%
FW Dataset	83.24%	80.72%

Table 8: Test Accuracies of Classification on ACS and FW Datasets

4 Discussion

As shown in 8, we got the results that the accuracy of classifying FW dataset was much better than the accuracy of classifying ACS dataset when we used the same procedure and the same model, consistent with [7] and [3]. After we checked the data samples, we found that the samples in ACS dataset were much "noisier" than those in FW dataset. As shown in 3, some sample images contain only a part of clothes which makes it really hard to tell what category it belongs to, and some samples are basically not images of clothes, the main parts of which are other objects. Besides that, the color of clothes in some images is too close to the background color or the outline is not clear in the original image, then after converting these images, it is much more difficult to detect the edges of clothes and then classify them correctly.

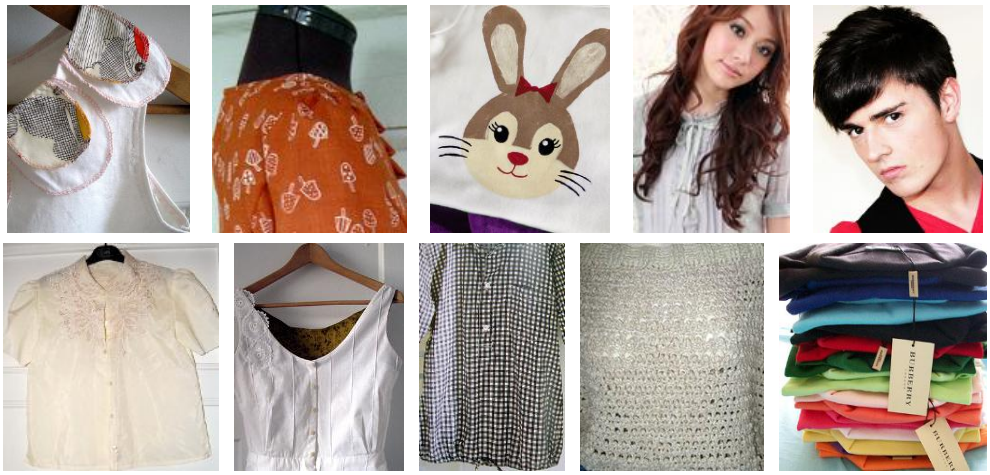


Figure 3: Examples of "Noisy" Images in ACS Dataset

On the other hand, we still got quite good results for FW dataset, close to [3], after we converted the sample images to 32 x 32 grayscale images. Obviously, the training time was much shorter. So, for clothing type classification or other similar problems, if the outline of target objects is the key point to do the classification, then we can first resize samples to a smaller size and then train the model, which can still achieve quite good results but take much shorter time. Also, this explains why our models did not perform well for ACS dataset and why the results were not as good as the classification results of Fashion MNIST dataset.

5 Conclusion

In our project, we did classification of Fashion MNIST dataset [1], by using Perceptron, Ridge Regression, LASSO, Logistic Regression, KNN, SVM, and CNN. Among them, CNN outperformed all the other methods, by achieving 94.67% accuracy on average, even though only a simple CNN model was built and trained. We also did classification of ACS dataset [2] and FW dataset [3] by using VGG11 [5] and AlexNet [4] with modified structures respectively. Inspired by [1], we converted samples images in those datasets to smaller grayscale images before training the models. We presented and analyzed the results as well. In conclusion, for Clothing Type Classification, usually the outline of clothes is the key point, i.e. we can achieve quite good classification performance by only considering the general outline of clothes but ignoring other attributes such as color, texture and so on. Therefore, we can convert sample images to a smaller size such that the outlines of clothes are still preserved, and then the training time can be largely reduced while quite good classification accuracy can still be achieved.

References

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

[2] C. Leistner C. Wengert T. Quack L. Bossard, M. Dantone and L. V. Gool. *Apparel Classification with Style*, pages 321–335. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[3] V. M. Pramod A. Subramanya, P. Srinivas and S. S. Shylaja. *Classification of Fashionwear Using Deep Learning*, pages 605–611. Springer Singapore, Singapore, 2018.

[4] I. Sutskever K. Alex and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[6] Y. Bengio Y. LeCun, L. Bottou and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[7] B. Lao and K. Jagadeesh. Convolutional neural networks for fashion classification and object detection.

[8] F. Rosenblatt. The perceptron–a perceiving and recognizing automaton. *Cornell Aeronautical Laboratory*, 1957.

[9] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–88, 1996.

[10] D. R. Cox. The regression analysis of binary sequences. *Royal Statistical Society*, 20(2): 215–242, 1996.

[11] N. S. Altman. An introduction to kernel and nearest-neighbour nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[12] V. Vapnik C. Cortes. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[13] Y. LeCun. Generalization and network design strategies. Technical report, University of Toronto, 1989.