Approach for Fashion Style Recognition on JD AI Fashion-Challenge

1st Huanghao Ding
SiChuan, China
dinghuanghao@gmail. com

2nd Wan Hui
SiChuan, China
wanhui0729@gmail. com

***Abstract*—the fashion style recognition subproject of JD AI Fashion Challenge is a multi-label classification task on fashion style, in which the sample labels are extremely imbalanced. We adopt up-sampling and random down-sampling to balance the distribution of labels and proposed a method on color changing of clothing to augment data, producing about twice as many new samples. In the training phase, the pre-trained models of Keras are used to make fine tuning, and to speed up the training, fine tuning in multiple phases and cyclical learning rate are also adopted. In the evaluation phase, we use greedy search to find the best threshold. And then, the performance is further improved by multi-level ensemble and our team ranks the first with the F2 score of 0.6834.**

***Index Terms*—fashion style, multi-label , deep learning**

I. Introduction

With the expansion of the global fashion consumer market and the popular adoption of AI technology in consumer sector, the combination of AI and Fashion is receiving more and more attention [15, 16, 17, 18, 19]. Fashion style recognition is a technology to automatically recognize the style of clothing through AI technology. The combinations of costumes are ever-changing, and each combination belongs to one or more styles. And unlike simple item identification, fashion style recognition requires strong professional knowledge. Thus, fashion style recognition is one the most difficult multi-label classification task that combines fashion and AI technology.

The fashion-style recognition subproject of JD AI Fashion-Challenge has a total of 54,908 training samples and 10,000 test samples. Due to the relatively small number of training samples and the imbalanced labels, performance of each label to train a single neural network is not satisfactory. We train multiple labels simultaneously through the way of multi-task learning, which not only reduces the learning cycle, but also improves the F2 score of small-scaled labels.

There are mainly two ways to avoid the imbalanced labels. One way is to change the distribution of data through data generation, up-sampling and down-sampling and etc. And the other is to improve loss function by adopting weighted loss function, focal loss [1] and etc. After experimentation, we combine data generation, up-sampling and down-sampling to generate new datasets to reduce the label imbalance. We use different backbone network to train a great number of single-models through transfer learning, and improve the diversity of the model by parameter disturbance. At last, the top-N models of each label with correlation less than 90% are selected for ensemble. The way to integrate is using XGBoost [12] based stacking [2] and vote based bagging [3].

II. Architecture and Training

A. Architecture of Network

Architecture of network used in this paper is pre-trained CNNs with customized full connected layer. The pre-trained CNNs mainly originate from the open source project of Keras [4]. The range of newly added hidden fully connected layer is in [1, 3], with neurons in [128, 1024]. And in most cases, the newly added hidden layer should be 1 with 256 neurons. We initialize the neurons with xavier initialization [5], and adopt batch normalization (BN) [6] between each newly fully connected layer and the ReLU [7] activation. The numbers of neurons in the output layer is the same as that of labels, with binary cross-entropy loss. Adopting such structure means constructing a learning network for multi-task, and enabling different labels to share the CNNs in the bottom layer while improving the performance of small-scaled labels.

B. Data Augmentation

We follow the practice in [8, 9]. Performed random cropping and horizontal flip on the image. In addition, stretching and compression within 10% of the side of the image can further improve performance.

The color of the clothing is diverse, and the same style of clothing often has a series of colors. The existing color augmentation technology such as PCA Jitter [8] only performs color changes within small extent and with some certain randomness that can result in strange colors. However, segmenting the clothing first and then changing the colors of clothing area also tends to make some mistakes in the edge due to limited accuracy of image segment algorithm.

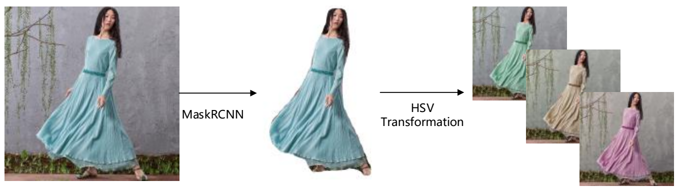

Fig. 1. Changing the colors of clothing.

To make up the disadvantages and limits mentioned above, we propose the following methods: (1) Adopting the relatively advanced Mask-RCNN network for portrait segmentation; (2) Inferring the colors of clothing based on the portrait area; (3) Performing HSV change of colors in the full image. The advantage of this method is that there are no cutting edges. Nevertheless, there is also an unsatisfactory side. That is, when the backgrounds and clothing enjoying the same colors, the colors of the backgrounds will also take some changes. In order to reduce the impact of such errors, we do not perform color changing on clothing with the similar skin colors. The effect is shown in Figure 1.

We generate about 130k new samples using the above methods. However, taking the computing capability and the label imbalance, we finally use about 10k samples corresponding to small-scaled labels.

In the test phase, we just flipped the image horizontally and then average them predictions (10-crop testing [8] can be better, but it takes too much time).

C. *Imbalanced Label Processing*

In the JD AI Fashion-Challenge, the distribution of labels is extremely imbalanced. As shown in Table 1, the maximum number of labels (label 2) is nearly 480 times the minimum labels (label 0). We solve this problem by up-sampling and down-sampling. Up-sampling is divided into two types. First, performing color changes to the small-scaled samples. Second, performing simples repetitions (reduce overfitting by randomly augmented during the training phase). Since a single sample while associated with a set of labels simultaneously, the up-sampling method will lead to the increase of samples corresponding to large-scaled labels. Overall, however, the ratio difference between the two can be reduced. And in the down-sampling method, only samples with relatively large-scaled labels are deleted randomly. In order to increase the random factors in the model, each epoch samples are deleted in random. Furthermore, we also try focal loss and weighted binary cross-entropy loss, but neither of them performes well.

TABLE I. SAMPLE DISTRIBUTION

Label	0	1	2	3	4	5	6
Quantity	104	4292	49893	262	18722	8882	197
Label	7	8	9	10	11	12	
Quantity	1356	49389	7094	3712	1249	360	

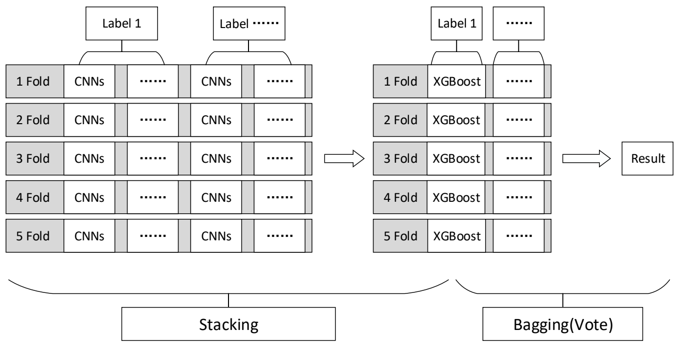

Fig. 2. Model Ensemble .

D. *Optimizing Parameters*

In CNNs training phase, we use the default parameters of Adam [9] with a mini-batch size of 32 (due to memory issues, when use DenseNet201 [10] and large resolution image, the batch size is 16). We adopt cyclical learning rate (CLR) [11] and periodical fine tuning (the first few layers of the network are trained at a large learning rate, and then the learning rate is gradually reduced and the number of layers trained is increased, and don't training a small part of the network under the bottom) to speed up training. In JAFC fashion style recognition datasets, training a model used less than 8 epochs. The image is resized with its shorter side range in [224, 400], the larger the image size, the better the effect. But considering the computing capability,we mainly use the size of 224.

During the XGBoost training phase, we combine 4 *eta* ranging from [0.05, 0.1, 0.15, 0.2] with 9 *max_depth* from [2, 11) and 5 *min_child_weight* from [1, 5]. Through early stopping technology, a series of models are obtained for each label, and we can select the optimized one according to F2 score on cross-validation sets. To save the computing time, we assign parameters to multiple computers for training and then perform combination.

E. *Threshold Search*

Both CNNs and XGBoost use a binary cross-entropy function and the output value is a float number between (0, 1). But the final output value should be a binary number, so we need to use a threshold for binaryzation. Usually, in traditional binary classification tasks, the threshold is 0.5. However, in the multi-label classifications, each label needs to be different on thresholds due to imbalanced labels. We can consider threshold search as an issue on optimization, which means to optimize F2 score on cross-validation sets by adjusting the thresholds. We adopt greedy algorithm that divides the threshold by 100 on 0 to 1 and then traverses to find the optimal threshold. We make comparison between this algorithm and the L-BFGS algorithm, The two are similary in F2 score, but the greedy algorithm is more quick and stable.

F. *Model Ensemble*

The key point of model ensemble is “performing well while maintaining difference”. To achieve this goal, we do the following works: (1) Adopting all of the pre-trained models in Keras; (2) Performing stochastic disturbance to datasets and parameters in the training process to increase the diversity of models; (3) Filtering out models with a similarity higher than 90% when finally selected models.

Since the optimal epoch corresponding to the multiple labels may be different in the same model training process. When selecting models, all the epochs of all models are uniformly ordered based on the F2 score for each label in the cross-validation sets, and the top-N models for each label are selected to ensemble. In order to prevent high similarity, among the multiple epochs only one epoch of a single model is selected.

The two-level model ensemble as shown in Figure 2 is adopted. The first level adopt stacking, and the second level bagging. In the training phase, 5-fold cross-validation is performed on the input images, and $13 \times 5 \times N$ CNNs models ($label \times fold \times top\text{-}N$) are trained. And the predictions of CNNs models could compose a set of new datasets. And then we continue to perform 5-fold cross-validation on this new datasets and train 13×5 XGBoost models ($label \times fold$). In the test phase, input each image into all CNNs models of 5 fold and generate 5 predictions. And then input each prediction into the XGBoost models of 5 fold, 25 predictions can be obtained. We average the output of the 5 XGBoost predictions of the same fold. Then, the final result can be obtained through voting on the averaged 5 predictions.

The label-based method to select and ensemble has an advantage that all trained models could be directly integrated without manual selection. Our best score adopt the top-20 CNNs models of each label to ensemble. After deleting the repetitive ones, there are totally 726 different CNNs models ensemble.

III. Experimentation

We compare the effects of different method used on the JAFC fashion-style recognition datasets with F2 score as criteria. To verify the effect of multiple backbones, all experiments are based on three models: DenseNet169 [10], ResNet50 [13], Xception [14]. And the F2 score of the basic model is shown in Table 2. Because each label is performed respectively to finish model selection and ensemble, the average F2 score is meaningless. As the biggest difficult of JAFC datasets is the label imbalance, without special instructions, all Tables only show the F2 score of label 0 (label with the minimum quantity of samples).

TABLE II. Label0 F2 Score of Backbone

Model	DenseNet169	ResNet50	Xception
Label0 F2 score	0.3846	0.3191	0.2368

A. *Color Changing of Clothing*

We perform color changing of clothing on pictures labeled 0, 1, 3, 5, 6, 7, 9, 10, 11, 12 with relatively less label quantity, as the data amount increased about 10K. The F2 score results after changing color are shown in Table 3. Although the effect of DenseNet169 is not improved, that effect of ResNet50 and Xception do get greatly improved.

TABLE III. Label 0 F2 Score

Model	DenseNet169	ResNet50	Xception
Label 0 F2 score	0.3719	0.3448	0.3

B. *Up-sampling*

The up-sampling is divided into two groups. We only make copying operations on label 0, which is the smallest scaled label. And the same operations are done to label [0, 3, 6, 12], which are relatively small in terms of scale. The copying multiple is divided into 10 times, 30 times, and 50 times, and the result is shown in Table 4. After using the up-sampling method, the performance of the three models is greatly improved.

TABLE IV. Label0 F2 Score of Up-sampling

Model	DenseNet169	ResNet50	Xception
$Label0 \times 10$	0.4459	0.4622	0.3595
$Label0 \times 30$	0.3817	0.4386	0.3097
$Label0 \times 50$	0.458	0.3571	0.3374
$Label[0,3,6,12] \times 10$	0.3318	0.3814	0.3448
$Label[0,3,6,12] \times 30$	0.3813	0.3077	0.3043
$Label[0,3,6,12] \times 50$	0.438	0.2479	0.3986

C. *Down-sampling*

Down-sampling reduces the four sets of samples with larger data volumes to the original 40%, 60%, and 80%. The result of F2 score is shown in Table 5. The large down-scaling will lead to performance degradation, which will further result in loss of useful data. However, the appropriate using of down-sampling also improves the performance.

TABLE V. Label0 F2 Score of Down-sampling

Model	DenseNet169	ResNet50	Xception
80%	0.3731	0.4262	0.3462
60%	0.2652	0.4124	0.25
40%	0.3125	0.3365	0.3179

D. *Integrating Multiple Technologies*

The F2 score of label 0 and the average F2 score of all the labels that integrates all above-mentioned technologies is shown in Table 6.

TABLE VI. Label0 F2 Score of Integrating multiple technologies

Model	DenseNet169	ResNet50	Xception
Label0 F2 score	0.4964	0.4687	0.5046
Average F2 score	0.6292	0.6081	0.6374

E. *Model Ensemble*

The results of integrating different quantities of models are shown in Table 7. It can be noted that, with an increase on the quantity of models, the final F2 score will also increase. But subjected to the size of basic models, the score will not increase after integrating the top 20 models.

TABLE VII. F2-SCORE OF ENSEMBLE

Top Model	Top5	Top10	Top20
F2 score	0.6720	0.6760	0.6834

IV. CONCLUSION

We train many models by adopting many technologies such as backbone networks, color changing of clothing, multi-phase fine tuning, up-sampling and down-sampling based on multi-labels, random data augmentation and etc. There are many differences among these models due to the change of parameters and the combination of technologies, especially among the small-scaled labeled models. We use model selection and ensemble that based on labels to take advantage of these diversities, and obtain good effects.

REFERENCES

[1] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[J]. IEEE transactions on pattern analysis and machine intelligence, 2018.

[2] Wolpert D H. Stacked generalization[J]. Neural networks, 1992, 5(2): 241-259.

[3] Breiman L. Bagging predictors[J]. Machine learning, 1996, 24(2): 123-140.

[4] Chollet F. Keras[J]. 2015.

[5] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks[C]//Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010: 249-256.

[6] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[J]. arXiv preprint arXiv:1502.03167, 2015.

[7] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines[C]//Proceedings of the 27th international conference on machine learning (ICML-10). 2010: 807-814.

[8] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[9] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.

[10] Huang G, Liu Z, Van Der Maaten L, et al. Densely Connected Convolutional Networks[C]//CVPR. 2017, 1(2): 3.

[11] Smith L N. Cyclical learning rates for training neural networks[C]//Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on. IEEE, 2017: 464-472.

[12] Chen T, Guestrin C. Xgboost: A scalable tree boosting system[C]//Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016: 785-794.

[13] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[14] Chollet F. Xception: Deep learning with depthwise separable convolutions[J]. arXiv preprint, 2017: 1610.02357.

[15] Jagadeesh V, Piramuthu R, Bhardwaj A, et al. Large scale visual recommendations from street fashion images[C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2014: 1925-1934.

[16] Xiao H, Rasul K, Vollgraf R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms[J]. arXiv preprint arXiv:1708.07747, 2017.

[17] Ma Y, Jia J, Zhou S, et al. Towards Better Understanding the Clothing Fashion Styles: A Multimodal Deep Learning Approach[C]//AAAI. 2017: 38-44.

[18] Al-Halah Z, Stiefelhagen R, Grauman K. Fashion forward: Forecasting visual style in fashion[J]. arXiv preprint arXiv:1705.06394, 2017.

[19] Liu Z, Luo P, Qiu S, et al. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 1096-1104.