Fashion popularity analysis based on online social network via deep learning

Jiachen Zhong, Rita Tse, Gustavo Marfia, Giovanni Pau

Jiachen Zhong, Rita Tse, Gustavo Marfia, Giovanni Pau, "Fashion popularity analysis based on online social network via deep learning," Proc. SPIE 11179, Eleventh International Conference on Digital Image Processing (ICDIP 2019), 1117938 (14 August 2019); doi: 10.1117/12.2539771

Event: Eleventh International Conference on Digital Image Processing (ICDIP 2019), 2019, Guangzhou, China

Fashion Popularity Analysis based on Online Social Network via Deep Learning

Jiachen Zhong[a, c], Rita Tse[a], Gustavo Marfia[b], Giovanni Pau[a, b, c]
[a]Macao Polytechnic Institute, Computing Program, Macao SAR; [b]Dipartimento di Informatica Scienza e Ingegneria, University of Bologna, Bologna, Italy; [c] UCLA Computer Science Department

ABSTRACT

In this paper, we provide an idea about how to utilize the deep neural network with large scale social network data to judge the quality of fashion images. Specifically, our aim is to build a deep neural network based model which is able to predict the popularity of fashion-related images. Convolutional Neural Network (CNN) and Multi-layer Perceptron (MLP) are the two major tools to construct the model architecture, in which the CNN is responsible for analyzing images and the MLP is responsible for analyzing other types of social network meta data. Based on this general idea, various tentative model structures are proposed, implemented, and compared in this research. To perform experiments, we constructed a fashion-related dataset which contains over 1 million records from the online social network. Though no real word prediction task has been tried yet, according to the result of dataset-based tests, our models demonstrate good abilities on predicting the popularity of fashion from the online social network using the Xception CNN. However, we also find a very interesting phenomenon, which intuitively indicates there may be limited correlation between popularity and visual design of a fashion due to the power and influence of the online social network.

Keywords: Fashion, Deep Learning, Social Network

1. INTRODUCTION

Nowadays, fashion plays a very important role in our lives. Fashion is a huge industry and creates plenty of commercial opportunities as well as risks. One of the major problems for fashion or apparel companies is predicting the popularity or quality of fashion design. This information is critical for business strategy decisions since it directly affects the price of the design and the quantity of sale. More importantly, if a company can forecast what will be popular next year, they may have an advantage over their competitor.

Although the scale of the fashion industry is huge, there are few studies focused on using machine learning model to predict fashion quality. This type of research is difficult to gain focus since the popularity of fashion is quite abstract and dynamic. Additionally, the lack of fashion-related data is also a huge problem. However, the combination of social networks and deep neural network yields a good opportunity for this type of study. Social networks can provide ample information, while deep neural network allows extracting high-level information from massive amount of data. Therefore, the idea of this research focuses on using an advanced machine learning model, deep neural networks, to predict the quality of fashion by training the model with data collected from a fashion-related social network.

2. RELATED WORKS

Like other subjects of study, the study of fashion in computer science, more specifically relating to computer vision and machine learning, can be summarized in a progressive level by level structure. According to our background study, there are three levels.

The first level of study focuses on clothing parsing which extracts the clothes in images. Basically, the focus at this level is similar to image segmentation. Edgar Simo-Serra et al. [1] used CRF (condition random fields) [2] model to perform this work. More recently Liu Ziwei et al. [3] [4] proposed the concept of fashion landmark and used CNN [5] (convolutional neural networks) to improve the performance on the fashion detection.

The second level is to extract some higher level attributes from the clothes in images, like the type, the style, the pattern, and even the texture of the clothes. In this level, the study focuses on the local detail of the visual representation of fashion. Liu Ziwei et al. [3] [4] used CNN to find attributes and utilize them to perform some benchmarks like clothes retrieval, clothes classification etc. Edgar Simo-Serra et al. [6] also used CNN to learn the style of the fashion in the pictures.

Eleventh International Conference on Digital Image Processing (ICDIP 2019) edited by
Jenq-Neng Hwang, Xudong Jiang, Proc. of SPIE Vol. 11179, 1117938 © 2019
SPIE · CCC code: 0277-786X/19/$21 · doi: 10.1117/12.2539771

In the third level, the research does not only focus on the fashion items themselves but tries to utilize those fashion items to conclude more abstract outputs, such as the occupation of people or the trends of the fashion. Zheng Song et al. [7] proposed an occupation prediction framework using sparse coding. Ziad Al-Halah et al. [8] evaluated and forecasted the fashion style by collecting the clothes purchasing data from Amazon (Amazon.com). Finally, Edgar Simo-Serra et al. [9] implemented a CRF model to predict the "fashionability" of the clothes of people using over 140 thousand data points collected from a fashion website. The third level of study has the closest relationship to our research work.

3. DATASET

We constructed a new fashion-related dataset which contains 1,109,182 user posts from a fashion-related social network, www.lookbook.nu. In this online social network, users are allowed to post their fashion photography and let other users to vote their posts. As a result, we collected the vote result (called hype on this website) for every post in our dataset. Other information were also collected for each post from this social network, for example, like how many fans the user has. The detail explanation and basic statistic information of all attributes in the dataset are presented in Table 1 and Table 2. Table 1 shows the basic statistical information of users' attributes, which may indicate the influence factor of users, from this social network. We only collected the information of users that are related to the 1,109,182 posts. Table 2 presents the basic statistics of posts' attributes from this social network. Among the four attributes of the posts, Hype, the number of positive votes given by other users, is the target attribute which we wish our model to predict.

Table 1. Basic statistics of the users

Attributes	SD.	Mean	Median	Max	Min	Top 1%
Fans	5148.23	1045.26	134	203438	50	21100
Looks	84.44	51.69	25	1962	0	404
Heart	293.10	44.92	6	20355	0	675
Karma	29673.09	6836.33	1381	964167	0	98500
Following	1170.47	229.89	61	120668	0	2970
Topics	2.54	0.23	1	214	0	4
Comment	4828.81	802.49	130	313853	0	10500
CKarama	239.81	21.14	0	14450	0	380
Views	58158.01	14620.08	5809	2825501	11	162000

Table 2. Basic statistics of the posts. SD. stands for Standard Deviation

Attributes	SD.	Mean	Median	Max	Min	Top 1%
Hype (vote)	262.12	147.28	70	9851	0	1320
UserVisit	3224.43	1619.40	812	302293	6	15400
TagNum	6.40	2.33	1	298	0	28
ItemNum	1.98	2.34	2	61	0	7

The entire dataset is randomly divided into three parts in the quantity of 800,000, 200,000 and 109,182 for Training, Test, and Validation respectively. We processed the entire dataset by blurring the Top 1% data which means the max value of the dataset is reset as the value at Top 1%, and those 1% data which exceeds the new max is also dealt as the new max value. After resetting the new max value and blurring data, each attribute is normalized into a range from 0 to 1. Moreover, all images were reshaped into 224 by 224 size via filling the black pixel to maintain the original shape of the image, and the RGB values of every pixel were also normalized into a range from 0 to 1. In addition, besides those data and images, we also collected the time-related data attributes, including post time, the time when user joined the social network, the time we fetched the post, and the last time we recorded the user information. The raw data are in timestamp format, but we transferred them into the number of months from when this website deployed, which is around March 2008. These time-related data were also combined with other non-image data as the meta data for machine training.

4. MODEL DESIGNS

4.1 General ideas

The intuitive idea of the model design is very simple: In order to better utilize the images, there should be some pre-processing. Following this direction, the model should contain two subcomponents: the Feature Extraction Model and the Regression Model. Figure 1 shows the workflow of this model. The model begins with the Feature Extraction Model which takes a fashion image as input and produces some representative features of the fashion. Combined with other meta data from the fashion social network, the fashion features are processed as the training input for the Regression Model which predicts the mark of the fashion after several training iterations. Based on this abstract idea, there are several possible implementation methods.

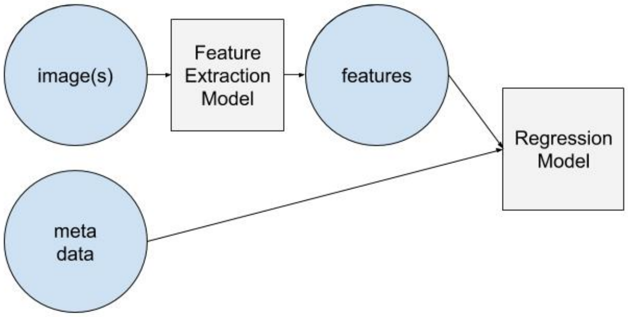


Figure 1. General idea of the model structure

4.2 Possible plans

One direct method is to build the Feature Extraction Model and the Regression Model separately. Another plan is to mix the two models and perform training simultaneously. There are more possible options. It is hard to determine which concept will perform better given that fashion task is so complex. Therefore, we compared several types of designs and then we chose the best performing model. We proposed three types of design approaches to try this task in general.

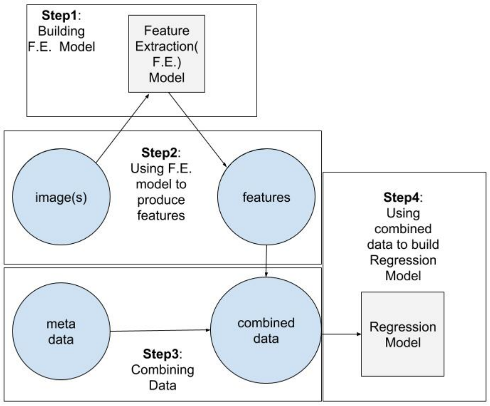

Step1: Building F.E. Model
Feature Extraction(F.E.) Model
Step2: Using F.E. model to produce features
image(s)
features
Step4: Using combined data to build Regression Model
meta data
combined data
Regression Model
Step3: Combining Data

Figure 2. Workflow of Type 1 models

Type 1: In this type, the Feature Extraction Model and the Regression Model are individual components. We first build the Feature Extraction Model and perform the useful feature extraction on images. Features will then be combined with the meta data to produce the training data for the Regression Model. The entire workflow is demonstrated in Figure 2.

Type 2: We may also deal with the Feature Extraction Model and the Regression Model as a whole component. In this case, the training targets on both the Feature Extraction Model and the Regression Model simultaneously in each iteration of the training. This is hard to perform in many machine learning approaches but it is feasible in the CNN which may be dealt as an end-to-end model and the convolution part of the CNN can be seen as feature extraction processing. Therefore, the entire workflow is demonstrated in Figure 3.

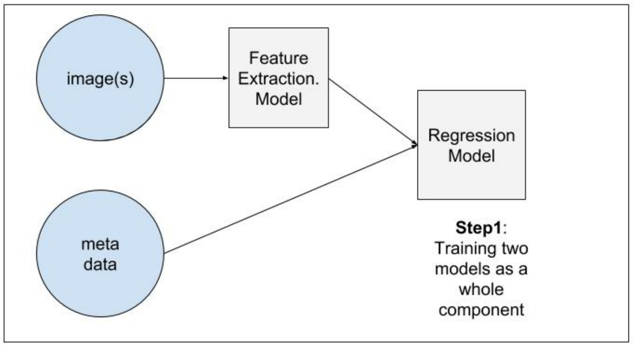

image(s)
Feature Extraction. Model
Regression Model
meta data
Step1: Training two models as a whole component

Figure 3. Workflow of Type 2 models

Type 3: Besides separating or integrating the Feature Extraction Model and the Regression Model, we may also separately process two kinds of input workflow. Instead of combining image features and meta data before inputting to the Regression Model, it is possible to allow meta data workflow and image workflow to perform regression separately before producing the final target output. Since the regression output target is also rescaled into a range from 0 to 1, using weights to recombine them into the final target output is feasible. After combining them by the weights which sum up to 1, we can perform the backpropagation on the two workflows simultaneously during the training. The entire workflow of this type is shown in Figure 4.

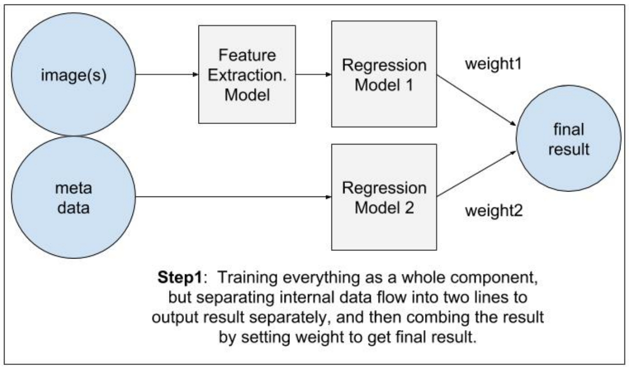

image(s)
Feature Extraction. Model
Regression Model 1
weight1
final result
meta data
Regression Model 2
weight2
Step1: Training everything as a whole component, but separating internal data flow into two lines to output result separately, and then combing the result by setting weight to get final result.

Figure 4. Workflow of Type 3 models.

4.3 Transferring general ideas into details

According to the general idea and three overview plans, 16 specific models were implemented in order to achieve the task as well as performing the comparison. The details of 16 models are explained in details in the following tables.

Table 3 shows the configuration of 3 structures in Type 1. In Type 1, since the Feature Extraction Model and the Regression Model are dealt as the two individual components, we need to assign the Feature Extraction Model with fashion-related feature extraction ability. Here, a well-collected fashion dataset, DeepFashion [3], was employed in order to bring the fashion-related feature extraction ability to the Feature Extraction Model which is a CNN in Type 1. The brief workflow is as: 1. Training a VGG-16[10] by using DeepFashion dataset. 2. Inputting our images to this trained VGG-16 to get the features. 3. Combining features with the social network meta data and using these data to train the Regression Model which is a MLP. We may select two types of features produced by the trained VGG-16. The first type of features is the output of the last convolution layers of VGG-16, which is a 7 by 7 by 512 numerical matrix and is unreadable to human-being. The second type of features is the final output of VGG-16 (output of fully-connected layers), which is a 50+16+16+1000 dimensions vectors. This feature is the training target of DeepFashion dataset including the 16 landmarks (location coordinates) and its visibility of fashion images, 50 fashion styles, 1000 fashion attributes; and this type of feature is readable to human-being. We also trained one more VGG-16 by only using the fashion landmarks part of the DeepFashion dataset as the Feature Extraction Model for comparison only; we only used the output of last convolution layer of this VGG-16 to produce the features.

Table 3. Detail Implementations in Type 1. F.E. denotes Feature Extraction. The value in parenthesis of the Features Column denotes the size and shape of the features. The value in parenthesis of the Regression Model Column denotes the layers and number of hidden units in the MLP (e.g. 2×2048 denotes 2 layers with 2048 hidden units in each layer).

No.	F.E. Model	Features	Regression Model
1	VGG-16 + Landmarks only	Output of last Convolution Layers (7×7×512)	MLP (2×2048)
2	VGG16 + Full DeepFashion	Output of last Convolution Layers (7×7×512)	MLP (2×2048)
3		Output of fully-connected Layers (50+16+16+1000)	MLP (2×2048)

There are 7 implemented models in Type 2. The detail configuration is shown in Table 4, and the basic idea of Type 2 is to utilize the advantage of the CNN to perform the automatic feature extraction during the training. Therefore, the key is the convolution layers structure. We selected 7 different types of CNN structures, VGG-16, ResNet Family [11] (ResNet18, ResNet34, ResNet50, ResNet101, ResNet152), and Xception [12], to achieve the task and perform the comparison.

Table 4. Detail Implementations in Type 2. F.E. denotes the Feature Extraction. The value in parenthesis after MLP denotes the layers and the number of hidden units in the MLP (e.g. 2×2048 denotes 2 layers with 2048 hidden units in each layer).

No.	F.E. Model + Regression Model
4	VGG-16 + MLP(2×2048)
5	ResNet18 + MLP(2×2048)
6	ResNet34 + MLP(2×2048)
7	ResNet50 + MLP(2×2048)
8	ResNet101 + MLP(2×2048)
9	ResNet152 + MLP(2×2048)
10	Xception + MLP(2×2048)

Finally, in Type 3, we allow meta data workflow and image workflow to perform regression separately to produce a value from 0-1 and combine them by their weights to produce the final target value. In this type, the weights are the key hyper-parameters to the model. Therefore, we used the same CNN model with different weights to perform the benchmarks for comparison. There are two extreme cases which make zero weight to the image or meta data flow to perform the comparison. We also designed one special case which puts the weight also as the model parameters waiting to be optimized. This can be done by adding one extra two hidden units layer as the MLP before the output. The hidden layer only contains two weights parameters waiting to be optimized. These two parameters are normalized by Softmax to make sure they sum up to one. The following formula simply shows the detail:

$$Final\ Output = \frac{f_1.e^{w_1}+f_2e^{w_2}}{e^{w_1}+e^{w_2}} \quad (1)$$

where e is the base of the natural logarithm, f_1 and f_2 are the image and meta output flows (two values between 0 and 1) from previews layer, w_1 and w_2 are the two weight parameters waiting to be optimized. Therefore, the actual weights for combining f_1 and f_2 are $\frac{e^{w_1}}{e^{w_1}+e^{w_2}}$ and $\frac{e^{w_2}}{e^{w_1}+e^{w_2}}$ respectively. We initially set the two weights, w_1 and w_2, to be equal, and let the model learn w during the training. The details of Type 3 models are shown in Table 5.

Table 5. Detail Implementations in Type 3. F.E. denotes the Feature Extraction. The value in parenthesis after MLP denotes the layers and number of the hidden units in the MLP (e.g. 2×2048 denotes 2 layers with 2048 hidden units in each layer).

No.	F.E. Model + Regression Model	Weight
11	Images + ResNet50 + MLP(2×2048)	1.0
	Meta Data + MLP(2×1024)	0
12	Images + ResNet50 + MLP(2×2048)	0.5
	Meta Data + MLP(2×1024)	0.5
13	Images + ResNet50 + MLP(2×2048)	0.1
	Meta Data + MLP(2×1014)	0.9
14	Images + ResNet50 + MLP(2×2048)	0.01
	Meta Data + MLP(2×1014)	0.99
15	Images + ResNet50 + MLP(2×2048)	0
	Meta Data + MLP(2×1014)	1.0
16	Images + ResNet50 + MLP(2×2048)	auto (initializing 0.5)
	Meta Data + MLP(2×1014)	auto (initializing 0.5)

5. EXPERIMENTS AND EVALUATIONS

5.1 General Experiment Setting

To perform a better comparison, all models are set with the same configuration except some hyper-parameters may be changed for exploring better performance. In this research, Batch Normalization [13], Drop Out [14], L2 Regularization [15], Mean Square Error (MSE), Loss Function and Xavier initialization [16] were applied to all models with the same setting. Adam [17] and Momentum [18] optimizer were employed for optimization. The activation function of the convolution layers and fully-connected layers were ELU [19] and Tanh (hyperbolic tangent) respectively. The Batch Size and the Learning Rate were different from model to model in order to achieve better performance. Training data were randomly shuffled after each training epoch. All models were trained by using TensorFlow [20]. We did not perform an exhaustive search for hyper-parameters. Therefore, the performance of models may have room for improvement.

5.2 Evaluation Method

In order to perform the comparison, three types of quantitative criteria were used. The first two are the loss function, MSE, and Mean Absolute Error (MAE) whose mathematic formulas are:

$$MSE = \frac{\sum_i^n (x_i - y_i)^2}{n} \tag{2}$$

and

$$MAE = \frac{\sum_i^n (x_i - y_i)}{n} \tag{3}$$

where x denotes the prediction of the model, y is the corresponding ground truth, and n is the number of samples in the test set.

The second one is the accuracy rate. However, because the model is performing regression, it is hard to compute the accuracy rate directly. In this research, we resolved this problem by setting a Tolerance Range(TR) to be the difference between the model output and the ground truth. Since we rescaled the output value to a range from 0 to 1. The TR we set for the evaluation is 0.05 and 0.10. It can be simplified into the mathematic formula (4).

$$a(x) = \begin{cases} 1, & if\ |x-y| < TR \\ 0, & otherwise \end{cases} \tag{4}$$

where *a(x)* denotes the accuracy of a single sample, *x* is the prediction output of the model, *y* is the ground truth, and *TR* is the value of Tolerance Range which we used 0.05 and 0.10 in our experiments.

And the overall accuracy of the test set can be denoted as:

$$accuracy = \frac{\Sigma_i^n a(x_i)}{n} \tag{5}$$

where *n* is the total number of samples in the test set.

5.3 Model Performances

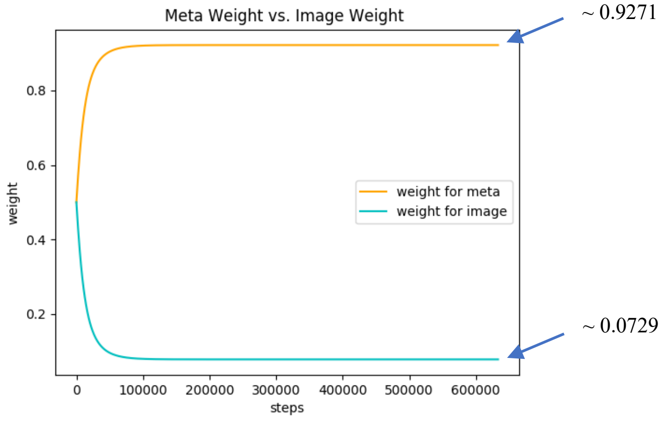
The performance results, which are based on the test dataset containing 200,000 records, of each model are presented in Table 6. Among all models, the Model No.15, which only utilized the meta data as the input, achieved the best result; whereas, the Model No.11, which only utilized the images as the input, achieved the worst result. In addition, among Type 1, the Model No.3 which used the full DeepFashion data to extract readable features achieved the best performance. Among Type 2, the Model No.10 with Xception CNN structure performed best. The model performance is acceptable for predicting an approximate the quality of fashion from the social network.

Table 6. Performance of each model

No.	**MSE**	**MAE**	**accuracy (TR=0.10)**	**accuracy (TR=0.05)**	**Settings**
Type 1 (Separated Training method)					
1	0.007489	0.065465	79.24%	49.20%	DeepFashion landmark
2	0.009117	0.071626	75.83%	45.83%	DeepFashion unreadable
3	0.007012	0.063026	82.33%	50.84%	DeepFashion readable
Type 2 (Mixed trained method with different CNN structures)					
4	0.007111	0.063206	81.59%	49.97%	VGG16
5	0.005405	0.055868	85.19%	54.93%	ResNet18
6	0.005500	0.055621	85.33%	55.86%	ResNet34
7	0.005512	0.055331	85.39%	56.55%	ResNet50
8	0.005318	0.054431	85.72%	56.96%	ResNet101
9	0.005192	0.054131	85.82%	57.25%	ResNet152
10	0.005092	0.053354	86.66%	57.90%	Xception
Type 3 (ResNet50 with different weights)					
11	0.022164	0.117171	51.33%	27.51%	Image only
12	0.005562	0.055931	85.21%	55.95%	Image:0.5, Meta:0.5
13	0.004512	0.050331	87.29%	60.25%	Image:0.1, Meta:0.9
14	0.004125	0.047783	89.73%	63.23%	Image:0.01, Meta:0.99
15	***0.003053***	***0.040886***	***92.97%***	***70.13%***	Meta data only
16	0.004325	0.048783	88.73%	61.23%	Image:auto, Meta:auto

5.4 Result Analysis

According to Table 6, it is strange in Type 1 that the performance of models drop down as the utilization of dataset increases. Model No.1 compared with Model No.2 takes less dataset but its performance is higher. In addition, Model No.3 compared with Model No.2 generates fewer features but gets better performance. Among Type 1, Model No.2 holds the most complex methods to utilize the images but achieves the worst result. On the other hand, in Type 2 models, it is obvious that the performance progressively improves by increasing the depth or complexity of the CNN structure. However, Model No.15 without using the image information outperforms all other models in all three types. The change of the weight in Model No.16 during training is illustrated in Figure 5. In this figure, the weight of image flow decreases as the weight of meta flow increases. The initial two weights($\frac{e^{w_1}}{e^{w_1}+e^{w_2}}$ and $\frac{e^{w_2}}{e^{w_1}+e^{w_2}}$) for two sides are both 0.5, but the image sides finally decreased to around 0.0729 and meta side increased to 0.9271. The model itself automatically wishes to give up the image side for a better fitting.



~ 0.9271

~ 0.0729

Figure 5. The change of meta data flow's and image flow's weight through training

We intuitively explained the strange result with two possible reasons. The first reason is that current Deep Learning approaches, those deep CNNs, are not completely suitable to extract useful information for fashion analysis. It may be possible that the fashion concept is too abstract for the CNN to understand. The second reason, which our team prefers more, is that the vote quantity of posts on the social network has limited correlation with the fashion popularity of images. Instead, it is more correlated with those meta social network data. Common sense is that getting others attention is much easier when a user is quite popular and has a lot of fans in a social network. This might be why after reducing the usage of images, the model may fit better since the users' influence powers are much more correlated to the vote than the image. Moreover, this situation might also be a true case in real fashion industry: A good advertisement is more important than hundreds of good fashion designs.

6. CONCLUSION

This research presented several designs and implementations on utilizing Deep Learning approaches with large amounts of social network data to judge popularity of fashion images. Comparisons between different design concepts were undertaken to explore more potential solutions. In addition, dataset based tests were done to show the utility of the model. Moreover, the experiment result pointed out a very interesting phenomenon that the popularity of fashion may have limited correlation with the visual design, whereas it is more correlated to popularity of the user. The behind reasons will be a potential area to explore. A large dataset related to fashion was collected in this research can facilitate other researches in the related topics. Real world prediction experiment will be one of the future work. We also may need to analyze the extracted features and to compare our methods with manually designed feature extraction method in the future. Since deep neural networks are a hot topic nowadays, there is no doubt that more deep learning research and applications in different areas will appear. This research is just an innovative and reasonable attempt to apply deep learning to an aspect of real life. We hope our work will inspire other teams or individuals to tackle this or other related challenging problems.

REFERENCES

[1] Simo-Serra, E., Fidler, S., Moreno-Noguer, F., and Urtasun, R., "A High Performance CRF Model for Clothes Parsing," Asian Conference on Computer Vision 9005, 64-81 (2014).

[2] Lafferty, J. D., Mccallum, A., and Pereira, F. C. N., "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," Eighteenth International Conference on Machine Learning 3, 282-289 (2001).

[3] Liu, Z., Luo, P., Qiu, S., Wang, X., and Tang, X., "Deepfashion: Powering robust clothes recognition and retrieval with rich annotations," IEEE Conference on Computer Vision and Pattern Recognition, 1096-1104 (2016).

[4] Liu, Z., Yan, S., Luo, P., Wang, X., and Tang, X., "Fashion landmark detection in the wild," In European Conference on Computer Vision, 229-245 (2016).

[5] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "ImageNet classification with deep convolutional neural networks," International Conference on Neural Information Processing Systems 25, 1097-1105 (2012).

[6] Simo-Serra, E., and Ishikawa, H., "Fashion Style in 128 Floats: Joint Ranking and Classification Using Weak Data for Feature Extraction," IEEE Conference on Computer Vision and Pattern Recognition, 298-307 (2016).

[7] Song, Z., Wang, M., Hua, X., and Yan, S., "Predicting occupation via human clothing and contexts," Computer Vision (ICCV), 2011 IEEE International Conference on, 1084-1091 (2011).

[8] Al-Halah, Z., Stiefelhagen, R., & Grauman, K, "Fashion forward: Forecasting visual style in fashion," arXiv preprint:1705.06394, (2017).

[9] Simo-Serra, E., Fidler, S., Moreno-Noguer, F., and Urtasun, R., "Neuroaesthetics in fashion: Modeling the perception of fashionability," IEEE Conference on Computer Vision and Pattern Recognition, 869-877 (2015).

[10] Simonyan, K. and Zisserman, A., "Very deep convolutional networks for large-scale image recognition," International Conference on Learning Representations (ICLR), (2015).

[11] He, K., Zhang, X., Ren, S., & Sun, J., "Deep residual learning for image recognition," Proc. of the IEEE conference on computer vision and pattern recognition, 770-778 (2016).

[12] Chollet, F., "Xception: Deep learning with depthwise separable convolutions," arXiv preprint arXiv:1610.02357v2, (2016).

[13] Sergey I. and Christian S., "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Proc. of the 32nd International Conference on Machine Learning, PMLR 37, 448-456 (2015).

[14] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., "Dropout: a simple way to prevent neural networks from overfitting," Journal of Machine Learning Research 15(1), 1929-1958, (2014).

[15] Krogh, A., & Hertz, J. A., "A simple weight decay can improve generalization," In Advances in Neural Information Processing Systems, 950-957 (1992).

[16] Glorot, X., & Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," Proc. of the thirteenth international conference on Artificial Intelligence and Statistics, 249-256 (2010).

[17] Kingma, D.P. and Ba, J., "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, (2014).

[18] Qian, N., "On the momentum term in gradient descent learning algorithms," Neural Networks 12(1), 145-151, (1999).

[19] Clevert, D.A., Unterthiner, T. and Hochreiter, S., "Fast and accurate deep network learning by exponential linear units (elus), " arXiv preprint arXiv:1511.07289, (2015).

[20] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., "Tensorflow: a system for large-scale machine learning," In OSDI 16, 265-283 (2016).