ICIC Express Letters
Part B: Applications
Volume **11**, Number **8**, August **2020**

ICIC International ©2020 ISSN 2185-2766
pp. 773–780

CLOTHING CLASSIFICATION USING CNN AND SHOPPING MALL SEARCH SYSTEM

Sunghwan Park, Yeryoung Suh and Jaewoo Lee*

Department of Security Convergence
Chung-Ang University
84, Heukseouk-ro, Dongjak-gu, Seoul 06974, Korea
*Corresponding author: jaewoolee@cau.ac.kr

Received January 2020; accepted April 2020

Abstract. *This study proposes a system that provides image-based clothing information services in the fashion online market environment among various service industries. This system learns data from CNN and creates a clothing classification system through SAREKnet and VGGNet model. Then, the color was found through k-means and the results were designed to be searched in the shopping mall. Unlike previous studies, we propose the method of using both image search and text search, to increase accuracy. In the future, it is necessary to study through the method of searching for whole-outfit, the method of deriving more class, and the algorithm that can automatically find clothes.*
Keywords: CNN, VGGNet, K-means clustering, Clothing classification, Image classification

1. **Introduction.** AI is a key driver of the 4th industrial revolution due to the development of deep learning that allows self-learning through given data by imitating human neural networks. It is significant in that AI can process big data collected from various data sensors such as IoT and create value on raw data. At present, AI is currently being applied in various fields such as the public, automobile, education, and finance.

In this study, we propose a system that provides image-based clothing information service in fashion online market environment among various service industries. The existing online fashion market provides customer recommendation through customer data. Most of these recommendation services have provided clothing search and recommendation through past purchases record of customers and text-based keyword searches [5]. However, there is a disadvantage in that the customer cannot find the desired image intuitively and must find the appropriate keyword in the system.

For this purpose, we have developed a system that classifies clothes, recognizes colors by using CNN (Convolutional Neural Networks) [6] structure, and then retrieves the information directly to the shopping mall. The data set used DeepFashion [1], which was reconstructed into a category that is frequently worn in Korea. We used CNN to classify clothes. We used SAREKnet and VGGNet [7] which were designed by us. We have developed a system for finding clothes using the k-means clustering algorithm [4] and searching for them in the shopping mall.

The image-based clothing retrieval system proposed in this study is expected to provide a more intuitive and highly customized service because it can find relevant products through the image if the user has a photograph of the desired clothing.

2. **Related Work.** The first part of this section deals with image-based search research. Next, we describe algorithms like artificial neural networks to be used in this paper.

DOI: 10.24507/icicelb.11.08.773

2.1. **Image-based search research.** In [5], Lim and Nang proposed a search result model and a classification result model to improve the existing method and proposed a method to combine each model. Through re-ordering, this paper supplemented the disadvantages of each model and suggested not only product search but also recommendation service.

Ng et al. [10] proposed k-means and watershed segmentation algorithm were integrated to improve the medical image segmentation. The existing algorithms have over-segmentation and sensitivity to false edge-problems. Therefore, [10] introduced k-means clustering to improve the algorithm to classify the image appropriately.

2.2. **Related algorithm.**

2.2.1. *CNN (Convolutional Neural Network).* CNN is a neural network mainly used for image recognition, and the filter designated by the user is convolutioned with the input image. CNN is composed of Convolution Layer, Activation Function, Max Pooling Layer and Fully Connected Layer. When convolution between images is performed, the entire image size is reduced, so the image size is maintained by using zero padding technology. In this paper, we will recognize the type of clothes through clothes images, so we will use CNN suitable for image recognition.

VGG16. VGG16 is a type of CNN pretrained through image of ImageNet. It has 16 layers and uses 224 × 224 sized image as input image. VGG16 is classified into 1000 classes, but we obtain output result using only 10 classes in Figure 1 of Section 3.1.

2.2.2. *Color discrimination using k-means clustering.* K-means clustering distributes all pixels of the image in a space according to the RGB value and clusters it, as shown in Figure 9. The RGB mean value of the clustering can be the representative color of the cluster, and it can be made into a spectrum like the last image. In k-means clustering, the meaning of k is divided into several clusters, and this paper set it as 5.

3. Main Results.

3.1. **Image preprocessing.** Dataset used DeepFashion: Attribute Prediction Dataset [1] is produced by Multimedia Lab of The Chinese University of Hong Kong. However, the previous dataset has too much data (289,222) and too many classes (5,621), so the work was carried out to reason 5,265 datasets and 10 classes according to the Korean clothing style. In addition, for high accuracy in learning, we also worked to identify images that were misclassified or error.

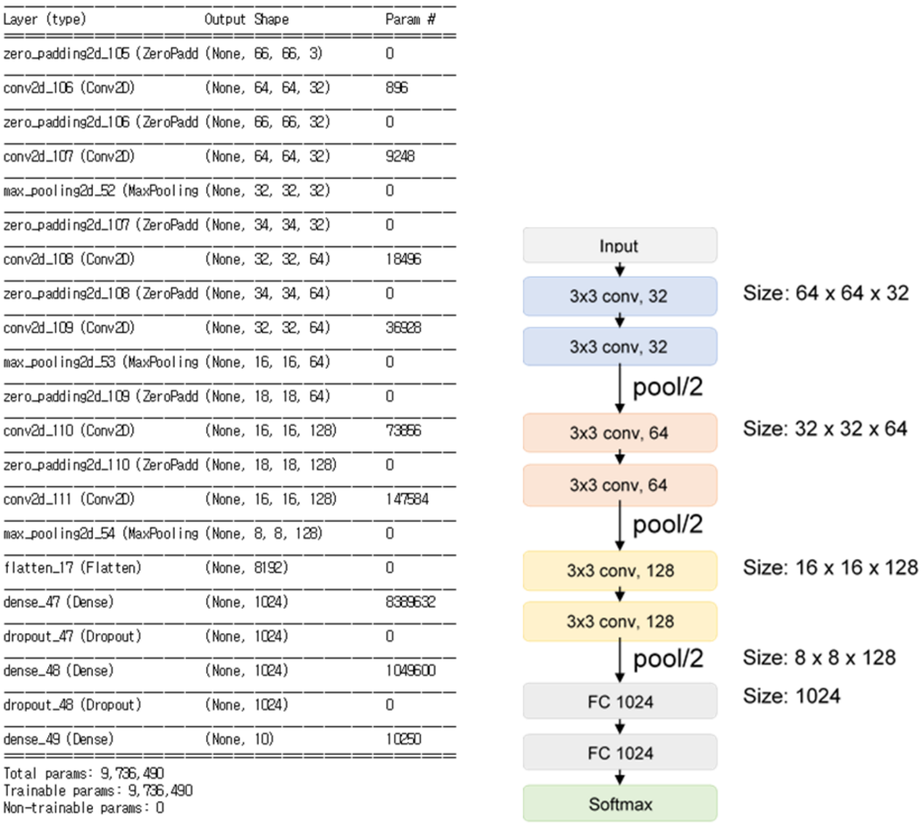
Cardigan	Coat	Dress	Hoodie	Jacket
Jeans	Leggings	Shorts	Sweatpants	Tee

Figure 1. Classes of preprocessed dataset

In this paper, each image is resizing according to the bounding box, and the characteristics of the clothes remain clear in the image. In order to apply VGGNet [7], the size of the image is resized to 224 × 224, which makes it an area of interest that is easy to recognize the characteristics of the clothes.

3.2. **Image attribute recognition using convolutional neural network.** In this paper, we implemented two CNN models using keras and openCV in GTX980ti environment. The first is SAREKNet, a self-made model. SAREKNet has an advantage for small computational complexity because it has an image size of $64 \times 64 \times 3$ while the image size of the VGGNet is $224 \times 224 \times 3$.

SAREKNet, as shown in Figure 2, is composed of zero padding, convolution, and zero padding and convolution again. Then Max pooling reduces the width and length of



```
________________________________________________________________
Layer (type)                 Output Shape              Param #
================================================================
zero_padding2d_105 (ZeroPadd (None, 66, 66, 3)         0
________________________________________________________________
conv2d_106 (Conv2D)          (None, 64, 64, 32)        896
________________________________________________________________
zero_padding2d_106 (ZeroPadd (None, 66, 66, 32)        0
________________________________________________________________
conv2d_107 (Conv2D)          (None, 64, 64, 32)        9248
________________________________________________________________
max_pooling2d_52 (MaxPooling (None, 32, 32, 32)        0
________________________________________________________________
zero_padding2d_107 (ZeroPadd (None, 34, 34, 32)        0
________________________________________________________________
conv2d_108 (Conv2D)          (None, 32, 32, 64)        18496
________________________________________________________________
zero_padding2d_108 (ZeroPadd (None, 34, 34, 64)        0
________________________________________________________________
conv2d_109 (Conv2D)          (None, 32, 32, 64)        36928
________________________________________________________________
max_pooling2d_53 (MaxPooling (None, 16, 16, 64)        0
________________________________________________________________
zero_padding2d_109 (ZeroPadd (None, 18, 18, 64)        0
________________________________________________________________
conv2d_110 (Conv2D)          (None, 16, 16, 128)       73856
________________________________________________________________
zero_padding2d_110 (ZeroPadd (None, 18, 18, 128)       0
________________________________________________________________
conv2d_111 (Conv2D)          (None, 16, 16, 128)       147584
________________________________________________________________
max_pooling2d_54 (MaxPooling (None, 8, 8, 128)         0
________________________________________________________________
flatten_17 (Flatten)         (None, 8192)              0
________________________________________________________________
dense_47 (Dense)             (None, 1024)              8389632
________________________________________________________________
dropout_47 (Dropout)         (None, 1024)              0
________________________________________________________________
dense_48 (Dense)             (None, 1024)              1049600
________________________________________________________________
dropout_48 (Dropout)         (None, 1024)              0
________________________________________________________________
dense_49 (Dense)             (None, 10)                10250
================================================================
Total params: 9,736,490
Trainable params: 9,736,490
Non-trainable params: 0
```

Figure 2. SAREKNet architecture

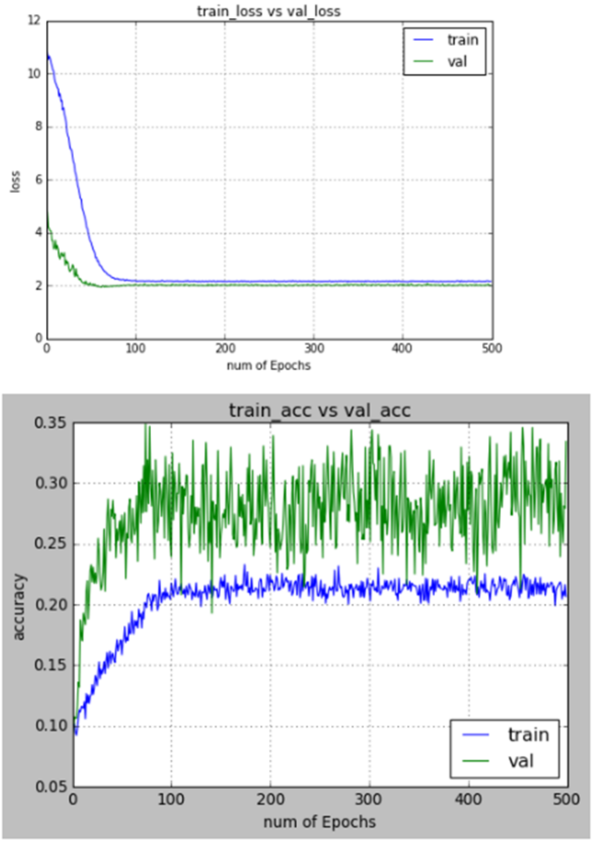
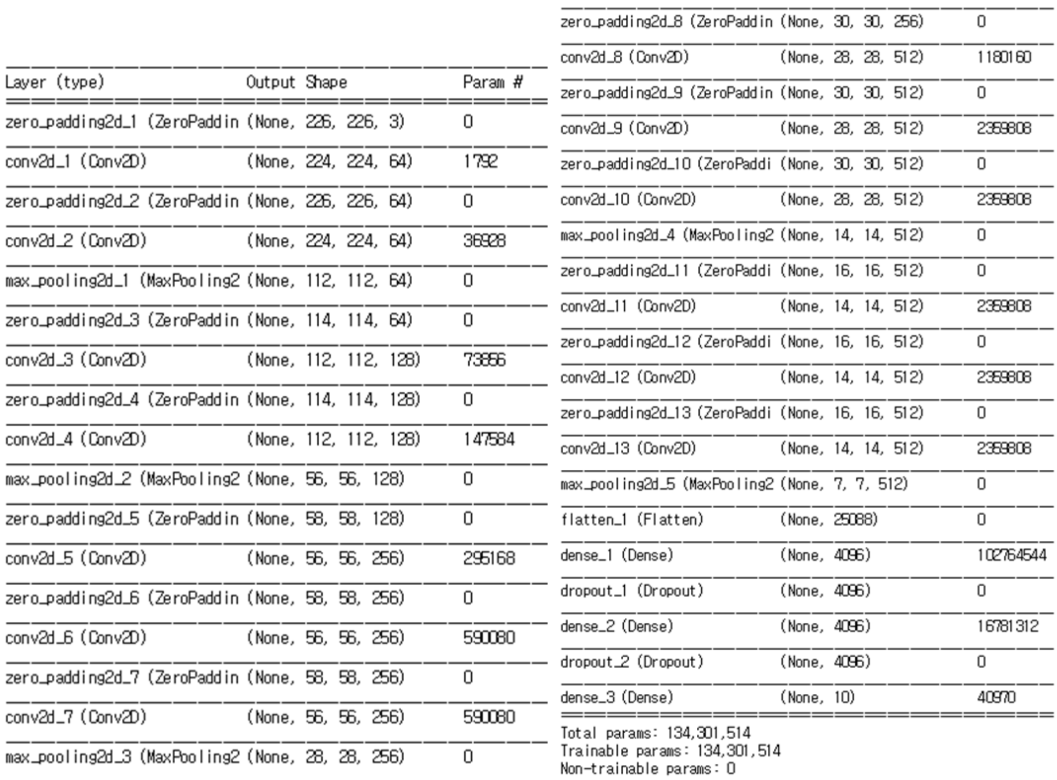


Figure 3. Result of SAREKNet

the image by half and repeats these three times in total. The number of parameters of SAREKNet is 9,736,490, which is less than the average CNN parameter value.

However, the results using SAREKNet have lower accuracy than general CNN as shown in Figure 3. The reason for this is that the VGG16 has 13 convolution layers, while only 6 convolution layers of SAREKNet are used.

The second model used is VGG16 of ImageNet. VGG16 was used by transfer learning weight with pretrained model. VGG16 has the same structure as Figure 4. Since it should have a size of 224 × 224 as an input image, it uses a preprocessed image made in the previous 3.1. It has 134,301,514 parameters, which is 13.8 times more than the previous SAREKNet, which requires more time than SAREKNet in training. However, because it learns through a larger number of convolution layers than SAREKNet, it has higher accuracy than SAREKNet as shown in Figure 5. However, the result of transfer learning without regularization is judged to be overfitting when the accuracy of the test dataset converges to 1 and the accuracy of the validation ('val' in Figures 5, 6, 7, and 8) dataset vibrates at about 75%. The overfitting result is difficult to apply to the actual dataset. The regularization used to solve this problem will be explained with the results in the following Section 3.3.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d_1 (ZeroPaddin (None, 226, 226, 3)       0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 224, 224, 64)      1792
_________________________________________________________________
zero_padding2d_2 (ZeroPaddin (None, 226, 226, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 224, 224, 64)      36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 112, 112, 64)      0
_________________________________________________________________
zero_padding2d_3 (ZeroPaddin (None, 114, 114, 64)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 112, 112, 128)     73856
_________________________________________________________________
zero_padding2d_4 (ZeroPaddin (None, 114, 114, 128)     0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 112, 112, 128)     147584
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 56, 56, 128)       0
_________________________________________________________________
zero_padding2d_5 (ZeroPaddin (None, 58, 58, 128)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 56, 56, 256)       295168
_________________________________________________________________
zero_padding2d_6 (ZeroPaddin (None, 58, 58, 256)       0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 56, 56, 256)       590080
_________________________________________________________________
zero_padding2d_7 (ZeroPaddin (None, 58, 58, 256)       0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 56, 56, 256)       590080
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 28, 28, 256)       0
_________________________________________________________________
zero_padding2d_8 (ZeroPaddin (None, 30, 30, 256)       0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 28, 28, 512)       1180160
_________________________________________________________________
zero_padding2d_9 (ZeroPaddin (None, 30, 30, 512)       0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 28, 28, 512)       2359808
_________________________________________________________________
zero_padding2d_10 (ZeroPaddi (None, 30, 30, 512)       0
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 28, 28, 512)       2359808
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 14, 14, 512)       0
_________________________________________________________________
zero_padding2d_11 (ZeroPaddi (None, 16, 16, 512)       0
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
zero_padding2d_12 (ZeroPaddi (None, 16, 16, 512)       0
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
zero_padding2d_13 (ZeroPaddi (None, 16, 16, 512)       0
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 14, 14, 512)       2359808
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 7, 7, 512)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_1 (Dense)              (None, 4096)              102764544
_________________________________________________________________
dropout_1 (Dropout)          (None, 4096)              0
_________________________________________________________________
dense_2 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dropout_2 (Dropout)          (None, 4096)              0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                40970
=================================================================
Total params: 134,301,514
Trainable params: 134,301,514
Non-trainable params: 0
```

Figure 4. VGG16 architecture

3.3. **Regularization.** Regularization tuned the result by adjusting the value of kernel_regularizer and active_regularizer provided by keras. L2 regularization was used for kernel_regularizer and L1 regularization was used for active_regularizer. The following Figures 6, 7, and 8 show the change of the loss value and the accuracy value when the values of kernel_regularizer and active_regularizer are 0.01, 0.08, and 0.05, respectively. As shown in Figure 8, it showed the most stable graph at 0.05 and used 0.05 when it produced actual results.

3.4. **K-means clustering.** If the image is divided into 10 classes through the neural network structure above, the color of the image is divided into the color of the image

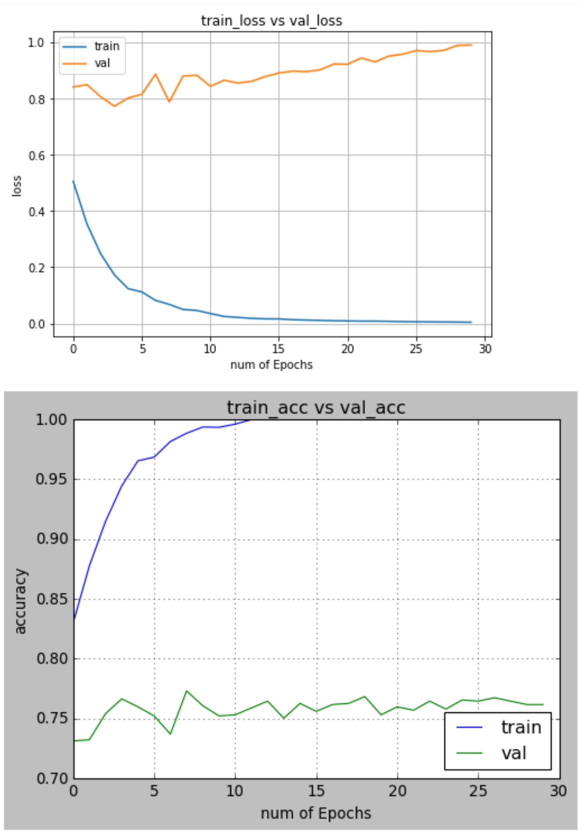


Figure 5. Result of VGG16 without regularization

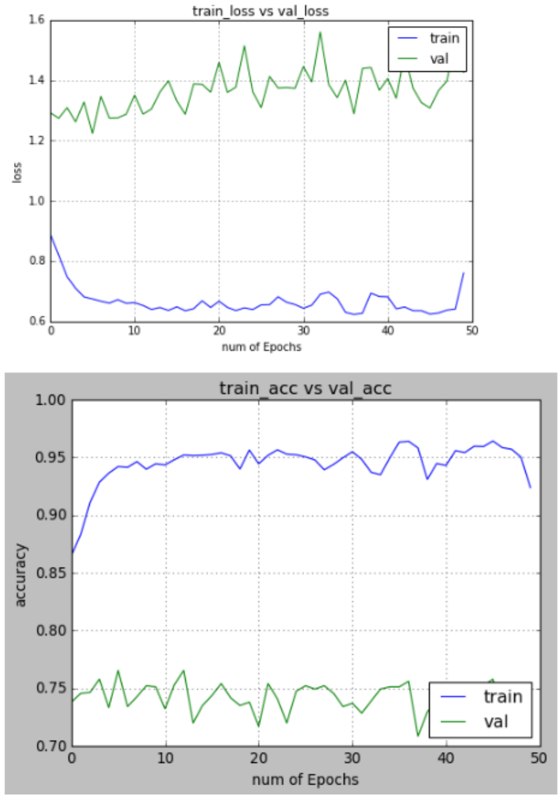


Figure 6. Using 0.01

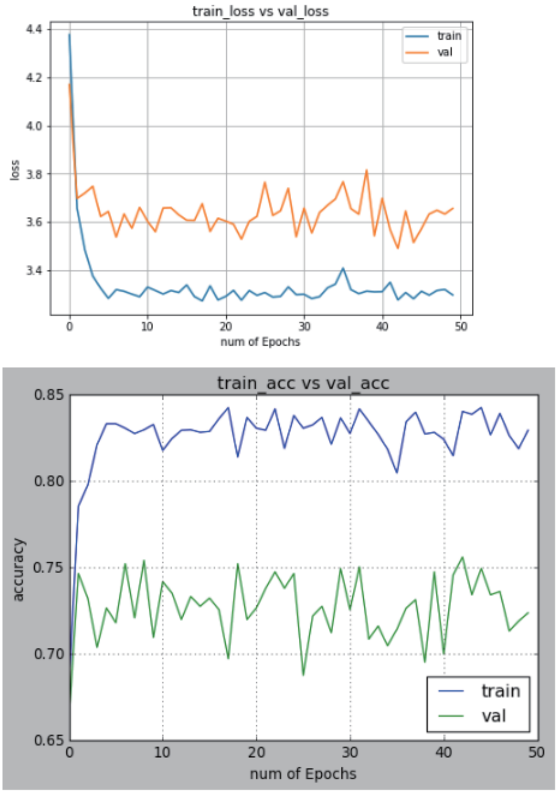


FIGURE 7. Using 0.08

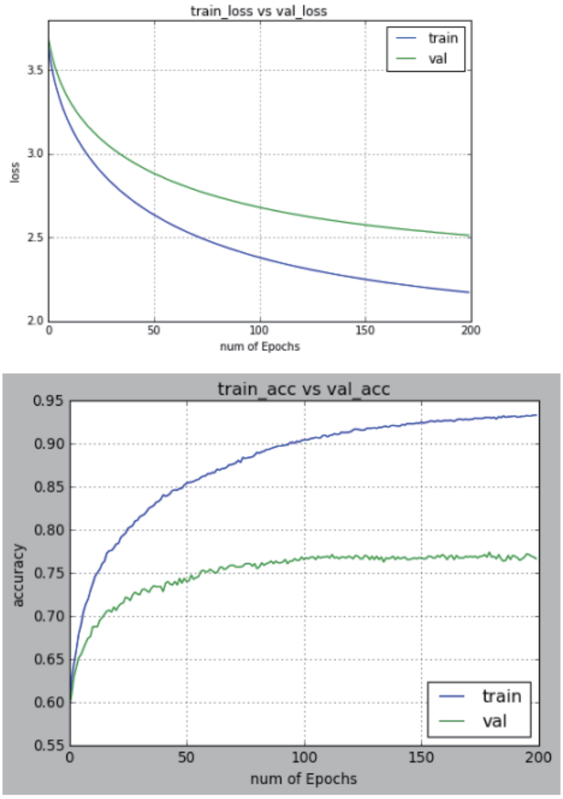


FIGURE 8. Using 0.05

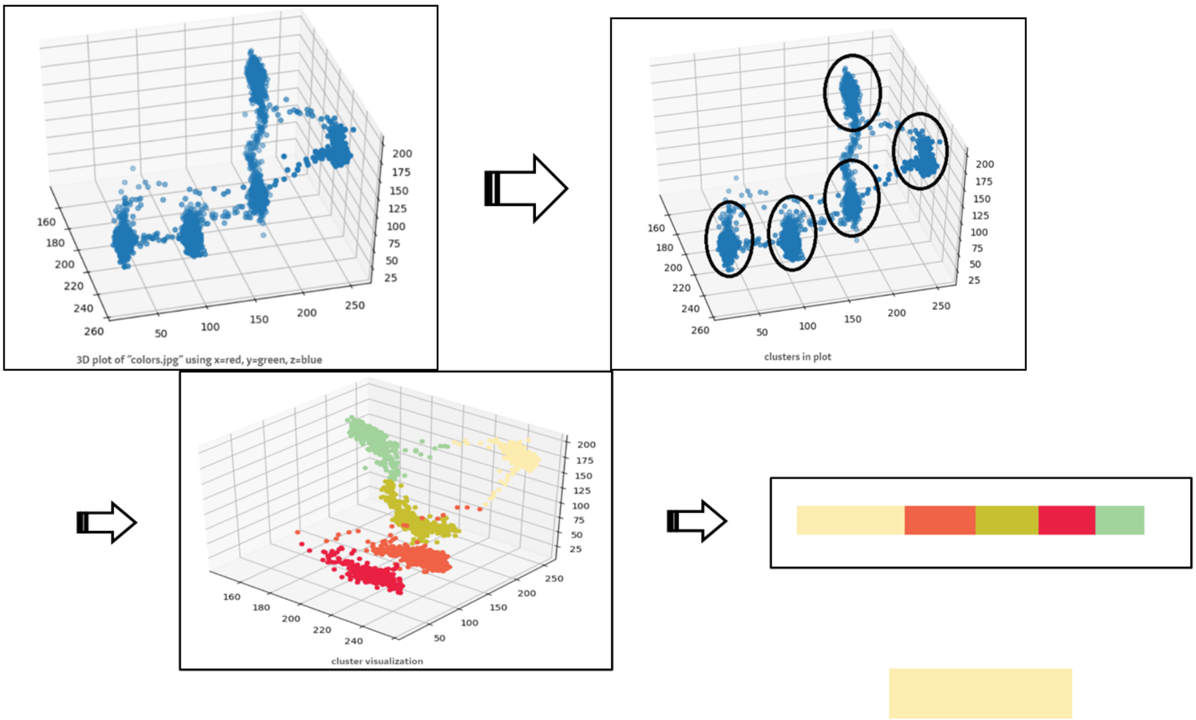
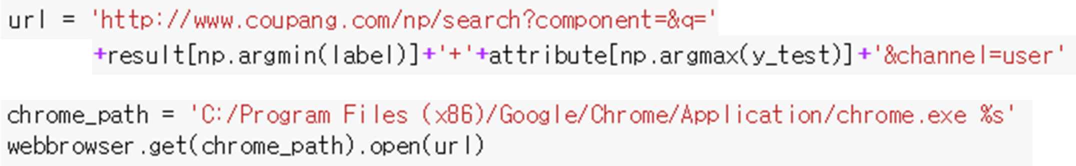


Figure 9. (color online) K-means clustering [4]

through k-means clustering, and the most color is represented by output. Dominant colors in an image using k-means clustering [4] by Thakkar were used and the results are as follows.

3.5. **Web searching.** Finally, we use the class value obtained by VGG16 model and the color value obtained by k-means clustering using the code described in Figure 10 to float the searched web. The site used for the search was "Coupang" site, a Korean distributor.



```
url = 'http://www.coupang.com/np/search?component=&q='
      +result[np.argmin(label)]+'+'+attribute[np.argmax(y_test)]+'&channel=user'

chrome_path = 'C:/Program Files (x86)/Google/Chrome/Application/chrome.exe %s'
webbrowser.get(chrome_path).open(url)
```

Figure 10. Code of web searching

4. **Conclusions.** In this paper, we design a system that classifies clothes using CNN structure [6] and recognizes colors and searches shopping malls immediately. The dataset was reconstructed to fit Korean style to increase the recognition rate, and the CNN structure, which is widely used in deep learning to recognize images, was used to recognize the attributes of clothes. Also, the color of clothes was recognized by using k-means clustering, and as a result, if only the image of clothes is input, the clothes are searched as similar as possible in the shopping mall based on the attributes of clothes and the color of clothes. The attributes and color-based search used in this paper is a different approach from the image search pursued in other papers. According to the results of Lim and Nang [5], the accuracy of the top 1 is up to 51% and the top 20 is over 75%. However, text-based search experiment in this paper can perform the search with a maximum accuracy of 77% with only one test image. Based on this, it is thought that the future research shows the possibility of research on the search method using both image search and text search.

Finally, future studies are to create an improved clothing classifier using fast R-CNN [8], faster R-CNN [9] and to recognize the shirts and pants at once without distinguishing them separately and to increase the number of classes of dataset.

Acknowledgment. This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2018-0-01799) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation).

REFERENCES

[1] Z. Liu et al., DeepFashion: Powering robust clothes recognition and retrieval with rich annotations, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.1096-1104, 2016.

[2] H. Chen, A. Gallagher and B. Girod, Describing clothing by semantic attributes, *ECCV*, pp.609-623, 2012.

[3] W. Di, C. Wah, A. Bhardwaj, R. Piramuthu and N. Sundaresan, Style finder: Fine-grained clothing style detection and retrieval, *CVPR Workshops*, pp.8-13, 2013.

[4] S. K. Thakkar, *Dominant Colors in an Image Using K-Means Clustering*, https://buzzrobot.com/dominant-colors-in-an-image-using-k-means-clustering-3c7af4622036, 2018.

[5] B. Lim, M. Jeong and J. Nang, A fashion image retrieval using cloth region proposal network based on deep learning, *KCC19*, pp.957-959, 2019.

[6] A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Advances in Neural Information Processing Systems*, vol.25, no.2, 2012.

[7] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, *ILSVRC*, 2014.

[8] R. Girshick, Fast R-CNN, *The IEEE International Conference on Computer Vision (ICCV)*, pp.1440-1448, 2015.

[9] S. Ren, K. He, R. Girshick and J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.39, no.6, 2015.

[10] H. P. Ng et al., Medical image segmentation using k-means clustering and improved watershed algorithm, *IEEE South West Symposium*, 2006.