International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 1 (2018) pp. 590-597
© Research India Publications. http://www.ripublication.com

Classifying Copyrighted Designs through Convolutional Neural Networks

Hye-Jin Kim and Yong-Hyuk Kim*

School of Software, Kwangwoon University, 20 Kwangwoon-ro, Nowon-gu, Seoul 01897, Republic of Korea.

ORCID: 0000-0003-3299-2932, 0000-0002-0492-0889
*(*Corresponding author)*

Abstract

Currently, research is being actively pursued on analyzing data gathered from social networks through machine learning and extracting meaningful information. In this study, we perform experiments to predict copyrights by training convolutional neural networks with image data gathered from Instagram, and we present a method which can improve model performance by changing various parameters. The results of performing experiments in which we predicted designs and related information in 10,000 collected images show a 116% improvement and 54% accuracy in experiments on classifying images into four classes through parameter tuning. Moreover, we confirm that we can expect positive improvements in the results if we perform the same method of tuning even when classes are added.

Keywords: Classification, copyrighted design, convolutional neural networks, machine learning

INTRODUCTION

In our current society, social networks have become one of the most important means of gathering information, and they are based on forming networks and sharing and communicating information between users. The typical examples of social networks are Instagram, Facebook, and Twitter. Twitter is used as a space for sharing personal opinions, but Instagram is a social network that specializes in image and video data, along with Facebook. The service started in 2010, and data can be shared with users who are within the scope of publicness set by the individual. One of Instagram's important features is the hashtag (#). Images can be described through hashtags, and they are mainly used for search purposes. In this paper, we use hashtag searches to collect images on Instagram which include copyrighted designs, and we use convolutional neural networks (CNN) to classify those images. In addition, our goal is to tune the CNN in various ways and perform experiments to find the method that is most suitable for the copyrighted data which has been gathered.

We have divided this paper into 5 sections, and the content of each section is as follows. In Section 2 we discuss previous research related to classifying social network images using convolution neural networks. In Section 3 we introduce the experiment design and the data we used, and we briefly describe our data collecting methods. In Section 4, we describe the process of implementing the experiments through several tuning methods. In Section 5 we analyze the experiment results, and in Section 6, we present our conclusions.

RELATED WORK

There is a variety of existing, previous research on using machine learning to classify image data collected from social networks. Huang et al. [1] combined images collected from Instagram and ImageNet data to create a dataset and used CNN to classify the images into categories. Kagaya et al. [2] performed research on using deep CNN for classifying images collected from Instagram into food pictures and pictures that are not food, and they obtained high accuracies of 96%, 95%, and 99% for their 3 datasets. Also, Singla et al. [3] used GoogleNet, which has an inception structure that combines the results of applying several filters of different sizes, to create a model which classifies image data gathered from social media into food pictures and pictures that are not food. Karayev et al. [4] performed experiments on recognizing image styles based on 165,000 images collected from Flickr and their entered tags.

There are also several typical studies on improving the performance of classification models through neural network tuning. Lin et al. [5] made detailed adjustments to a bilinear CNN model and analyzed the model's speed and accuracy. Kim [6] classified sentences using a CNN based on hyper-parameter static vectors. Also, they proposed a method of making simple modifications to the architecture so that all the static vectors for each task could be used. Girshick et al. [7] used a CNN model to detect objects in a PASCAL VOC data set, and if the labeled learning data was insufficient they were able to make detailed adjustments for each domain through supervised pre-learning assistance tasks to greatly improve performance. Chatfield et al. [8] confirmed several useful CNN-based characteristics which greatly reduced the dimensions of the CNN output layer without having a negative effect on performance. Xu et al. [9] developed a deep CNN model which can capture decomposition characteristics and implemented it to include 2 sub-modules which were trained in a supervised fashion through suitable initialization. In addition to this, Girshick [10] used a fast-region based CNN to improve speed-related performance in detecting objects, including training the VGG16 network 9 times faster than existing R-CNN and making the test speed 213 times faster.

ENVIRONMENTS FOR DESIGN CLASSIFICATION

In this study, our goals were to use 8,000 pieces of image data collected from Instagram to tune 4 various parameters to improve model performance. The data used in the experiments were images including copyrighted designs like the pictures in Fig. 1, and we modified and used the Google Chrome extension ChanHaRi[1] to collect the images. The collection program's running screen is shown in Fig. 2. The Chrome web driver was controlled remotely to search for user-specified hashtags on Instagram, Facebook, and Twitter. A Python module library was used to extract the image paths and automatically save images, text, and various information about the posted content on the user's local storage until the desired amount of data was collected. We used the collection program to collect 2,000 images from Instagram for each class. We created 4 classes for the experiments, and a total of 8,000 data items were collected. Table 1 shows information on each class.

Table 1. Class Information

Class	#Images
Minions	2,000
Stitch	2,000
Pikachu	2,000
Anpanman	2,000

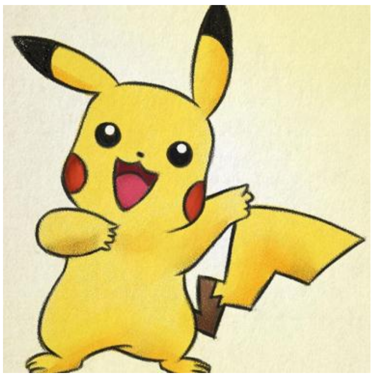
Each class is composed of typical copyrighted designs such as the Minions, which are a popular design from Illumination Entertainment. Sample images can be seen in Fig. 1. In this paper, we designed the experiments so there would be preprocessing on the 8,000 collected data items in which their suitability for use in the experiment was judged to limit the number of images and the existing image sizes were reduced. In the experiments, the preprocessed data was used to train the CNN. A CNN is an artificial neural network that uses an additional convolutional layer and a pooling layer. It uses few parameters, and it has the advantage of showing good performance for images and voices.

Software libraries for deep learning include Torch, Theano, Deepleaning4j, ND4J, NVIDA cuDNN, convnetjs, Caffe, and Tensorflow. Of these, we used Tensorflow[2], an open source machine learning library released by Google, to perform the experiments. Tensorflow is a Google machine learning library released as open source on November 9, 2015. It has the features of being able to use Python and C++ as programming languages and being able to run on a variety of platforms. Another feature is that Tensorflow is a library optimized for distributed processing, so it can operate on several CPUs and GPUs, and this helps with fast computation. Also, Tensorflow can operate on a TPU (Tensor Processing Unit) which is AI accelerated hardware developed by Google for even faster computation speed.

In the present research, we performed experiments in which we made changes to 4 parameters for learning. The parameters changed in the experiments included the type of neural network, activation function, weight initialization method, and number of layers. Also, we implemented an additional experiment in which the number of classes was expanded from 4 to 5, and we studied the effect the parameters had on the experiment when the number of classes was changed, just as we did when the number of classes was 4.

(a) Minions

(b) Stitch


(c) Pikachu

(d) Anpanman

Figure 1. Samples of copyrighted design data collected from Instagram

[1] *https://goo.gl/xdKT34*
[2] *https://www.tensorflow.org*

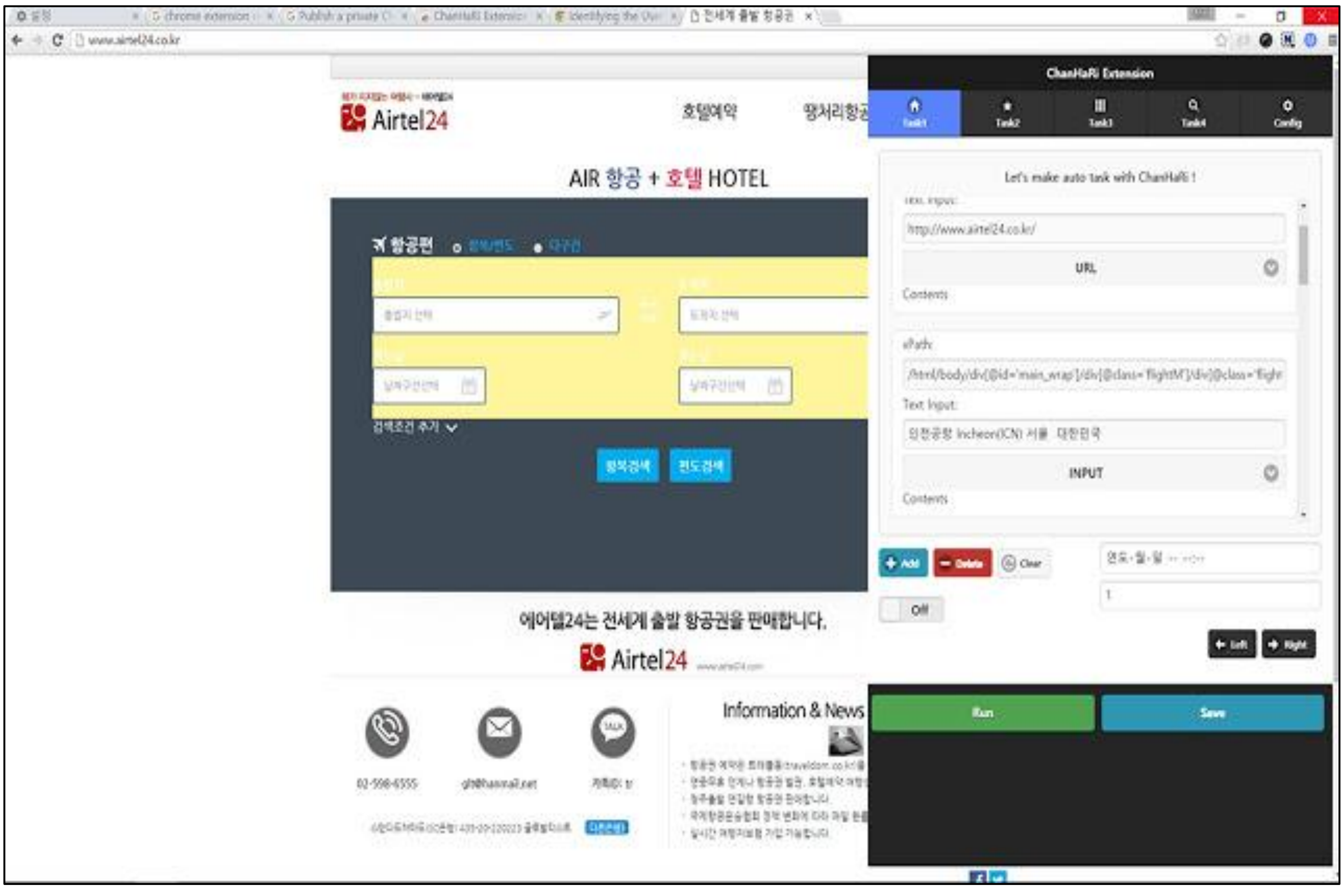


Figure 2. Example of running the ChanHaRi Extension used to collect data

NETWORK ARCHITECTURE

For the experiment data, we used hashtags on Instagram to collect a total of 8,000 images (2,000 for each class) with 640x640 pixel resolution. We determined whether the collected images could be used in the experiments and selected 800 suitable images to be used as data. The colors of the copyrighted design can act as an important factor in the experiment, so we kept the colors as they were and reduced the image sizes to 128x128 pixels.

Our experiments were implemented using Google's deep learning library, Tensorflow, and on a basic level they included 3 convolutional layers as in Fig. 3. Softmax regression, the regression method used in these experiments, is suitable for calculating probability. The evidence that the input data belongs to each class is numerically calculated, and then the calculated values are converted to probabilities to produce the results. The data sets are learned 100 to 1,000 times, and this is repeated 30 times to produce an average accuracy which is shown in Table 2. As can be seen in Table 2, the learning accuracy was the highest at 100 and 300 repetitions. Therefore, in our experiments we repeated the learning 100 times to achieve best learning at the fastest learning speed. Also, when a Deep Neural Network (DNN) was trained 100 to 1,000 times, and this was repeated 30 times to produce an average of the learning accuracy, the accuracy of all learning was the same at 0.65. Therefore, we performed a variety of experiments with a DNN at 100 repetitions.

The goal of the present research is to variously tune and test a neural network and find the most suitable method for the collected copyrighted data, so it was important to change 4 parameters when performing experiments. As mentioned before, the 4 parameters included the type of neural network, activation function, weight initialization method, and number of layers. For the first parameter, the type of neural network, we used CNN and DNN and performed experiments on each to compare them. The second parameter was the activation function. In our experiments, we compared using both the Sigmoid function and the ReLU function as the activation function. For the third parameter, the weight initialization method, we compared results when the Xavier initialization method [11] was used and when it was not used. Xavier initialization, a weight initialization method, selects a random number between the input value and the output value and divides it by the square root of the input value. The last parameter used in this study was the number of layers. In a CNN, that means changing the number of convolutional layers, and in a DNN, that means changing the number of hidden layers. We performed the experiment by changing the number of each layer type to 2, 3, and 5 layers.

Also, to confirm that the parameters had an effect on the experiment when the number of classes was changed just like when the number of classes was 4, we added a class which included images like those shown in Fig. 4 so that we were able to implement an additional experiment in which the number of classes was expanded from 4 to 5. For the expanded class, we gathered 2,000 images and extracted data from 200 of those images which were suitable. The additional data was included as shown in Table 3.

Table 2. Changes in accuracy according to number of times trained (training results)

Epoch	Accuracy
100	1.00
300	1.00
500	0.96
1,000	0.92

Table 3. Information on experiment with one expanded class

Class	#Images
Minions	2,000
Stitch	2,000
Pikachu	2,000
Anpanman	2,000
Hello Kitty	2,000

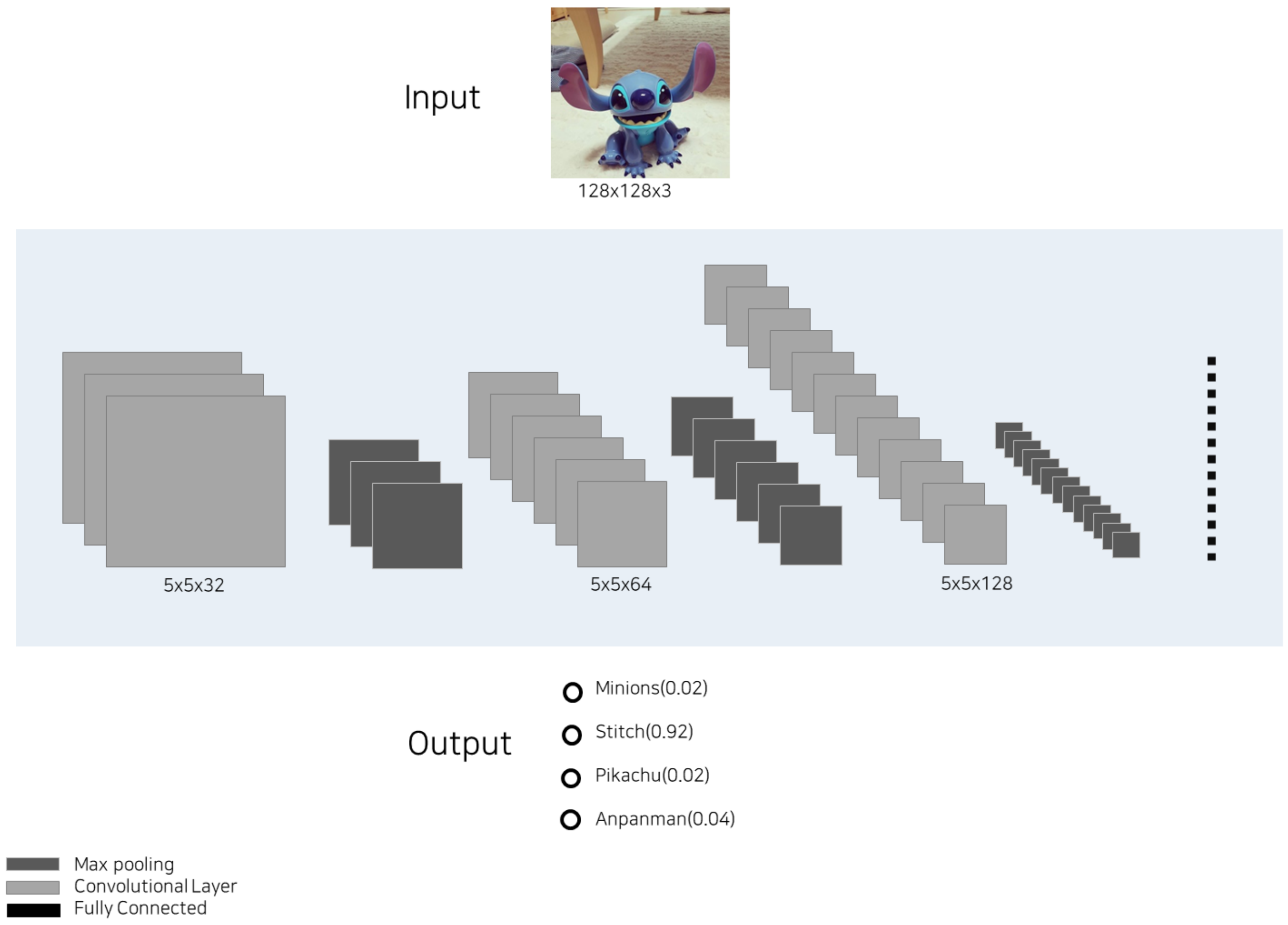


Figure 3. CNN structure that was used

Figure 4. Data sample (Hello Kitty) for experiment with expanded class

RESULTS

Performance of Basically Implemented Model

As a basis, this study used a network with 3 convolutional layers, the Xavier weight initialization method, and the ReLU activation function. Table 4 shows the accuracy, *F*-measure, precision, and recall that were the results of repeating 30 tests on the model created by learning 100 times using the CNN. Table 6 shows the prediction model's confusion matrix. Table 5 shows the average values from performing the experiment the same number of times with a DNN in order to make a comparison with the CNN. Table 7 shows the confusion matrix of the prediction model which was implemented using a DNN.

Table 4. Performance of basic prediction model using CNN

	Minions	Stitch	Pikachu	Anpanman
Accuracy	0.47			
Recall	0.61	0.42	0.37	0.55
Precision	0.50	0.50	0.47	0.47
F-measure	0.55	0.46	0.41	0.50

Table 5. Performance of basic prediction model using DNN

	Minions	Stitch	Pikachu	Anpanman
Accuracy	0.36			
Recall	0.40	0.47	0.47	0.38
Precision	0.45	0.36	0.39	0.35
F-measure	0.41	0.43	0.33	0.29

Table 6. Confusion matrix of CNN basic model

		Predicted			
		Minion	Stitch	Pikachu	Anpanman
Actual	Minion	61	8	15	16
	Stitch	24	42	12	22
	Pikachu	24	14	37	25
	Anpanman	13	17	15	55

Table 7. Confusion matrix of DNN basic model

		Predicted			
		Minion	Stitch	Pikachu	Anpanman
Actual	Minion	61	8	15	16
	Stitch	24	42	12	22
	Pikachu	24	14	37	25
	Anpanman	13	17	15	55

Model Results According to Activation Function

Based on the previous experiments' results, we performed experiments on the second parameter using a CNN. We compared the results of this study's second parameter, weight initialization method, according to the activation function. Table 8 shows a comparison of the training results of a model which uses Sigmoid as the activation function and a model which uses ReLU. Table 9 shows the performance of each implemented model. Also, Table 10 shows the confusion matrix of a model which uses Sigmoid and ReLU together.

Table 8. Comparison of training results of models with various activation functions

Activate Function	**Accuracy**
Sigmoid	0.25
ReLU	1.00
Sigmoid+ReLU	1.00

Table 9. Comparison of performance of models with various activation functions

Activate Function	**Accuracy**	***F*-measure**			
		Minions	Stitch	Pikachu	Anpanman
Sigmoid	0.25	0.00	0.40	0.00	0.00
ReLU	0.47	0.55	0.48	0.42	0.41
Sigmoid+ReLU	0.53	0.55	0.54	0.55	0.48

Table 10. Confusion matrix of CNN model using both Sigmoid and ReLU as activation function

		Predicted			
		Minion	Stitch	Pikachu	Anpanman
Actual	Minion	61	8	15	16
	Stitch	24	42	12	22
	Pikachu	24	14	37	25
	Anpanman	13	17	15	55

Model Results According to Weight Initialization Method

Based on the performance results according to the second parameter i.e. the activation function, we performed experiments on the third parameter. For this study's third parameter, the weight initialization method, we compare the results of whether or not the Xavier initialization method was used. Table 11 shows the results of a model that uses the Xavier initialization method and a model that does not. Table 12 shows the performance of each implemented model.

Table 11. Comparison of training results of models using various weight initialization methods

Weight Initialization	Accuracy
Random	1.00
Xavier	1.00

Table 12. Comparison of performance of models using various weight initialization methods

Weight Initialization	Accuracy	*F*-measure			
		Minions	Stitch	Pikachu	Anpanman
Random	0.45	0.56	0.41	0.44	0.48
Xavier	0.53	0.55	0.54	0.55	0.48

Results of Models with Various Numbers of Layers

The fourth parameter used in this study, the number of layers, refers to changing the number of convolutional layers in a CNN and changing the number of hidden layers in a DNN. Using the results of the previous experiments, we operated a CNN with ReLU as the activation function and the Xavier method as the weight initialization method. We performed the experiments by changing the number of layers to 2, 3, and 5 layers. Table 13 shows the results of repeating 100 rounds of training 30 times, and Table 14 shows the performance of each implemented model. Also, Table 15 shows the confusion matrix of the model with 5 convolutional layers which showed better accuracy than the basic model.

Table 13. Comparison of training results of models with various numbers of layers

Layers	Accuracy
2	1.00
3	1.00
5	1.00

Table 14. Comparison of performance of models with various numbers of layers

Layers	Accuracy	*F*-measure			
		Minions	Stitch	Pikachu	Anpanman
2	0.43	0.51	0.39	0.41	0.39
3	0.53	0.55	0.54	0.55	0.41
5	0.54	0.54	0.60	0.53	0.48

Table 15. Confusion matrix of model with 5 convolutional layers

		Predicted			
		Minion	Stitch	Pikachu	Anpanman
Actual	Minion	61	8	15	16
	Stitch	24	42	12	22
	Pikachu	24	14	37	25
	Anpanman	13	17	15	55

Results of Model with Classes Expanded to 5

In order to confirm that the parameters have an effect on the experiment when the number of classes was changed just as they did when the number of classes was 4, we expanded the number of classes from 4 to 5 and performed an additional experiment. In the additional experiment, we also changed several parameters, and Table 16 shows the corresponding training results. We quantified the *F*-measure, precision, and recall of the performance of the basic model which was expanded to 5 classes, and this is recorded in Table 17. The model's confusion matrix is shown in Table 18.

Table 16. Training results of various models with expanded class

Network	Activate Func.	Weight Init.	Layer	Acc.
DNN	ReLU	Xavier	3	0.57
CNN	Sigmoid	Xavier	3	0.20
CNN	ReLU	Random	3	1.00
CNN	ReLU	Xavier	3	1.00
CNN	ReLU	Xavier	5	1.00

Table 17. Performance of model with 5 classes and 5 convolutional layers

	Minions	Stitch	Pikachu	Anpanman
Accuracy	0.47			
Recall	0.61	0.42	0.37	0.55
Precision	0.50	0.50	0.47	0.47
F-measure	0.55	0.46	0.41	0.50

Table 18. Confusion matrix of model with 5 classes and 5 convolutional layers

		Predicted				
		Minion	Stitch	Pikachu	Anpanman	Hello Kitty
Actual	Minion	55	9	7	17	12
	Stitch	12	43	12	18	15
	Pikachu	18	8	41	19	14
	Anpanman	10	15	12	40	23
	Hello Kitty	17	12	23	17	31

CONCLUSION

We performed experiments on a method to predict copyrights using a CNN on designs included in Instagram images, and we looked at differences in performance according to changes in 4 parameters. The CNN model which was basically created to look at the effect of the first parameter had an accuracy performance of 47%, and the class which had the highest *F*-measure was the first class i.e. the Minions design by Illumination Entertainment. On the other hand, we found that the class which recorded the lowest numbers was the 3rd class, Pikachu. However, this was a better result than the 36% accuracy results of a DNN. The second parameter, the activation function, created large differences in both training and performance. We also found that using two activation functions can show better results than using only one function. In the experiments on the third parameter i.e. the weight initialization method, the performance of the model which used the Xavier initialization method was better by a small amount compared to the model that did not. The results of changing the final parameter i.e. the number of layers, confirmed that increasing the number of convolutional layers in a CNN was associated with performance improvements.

We also implemented experiments in which the number of classes was expanded from 4 to 5. We changed several parameters in the additional experiments as well. In the DNN, the training results and performance rapidly became poor when the number of classes was increased, and we were able to confirm limitations in the implemented model. On the other hand, in the convolutional neural network model, changes in the parameters showed similar effects to those seen in the existing model with 4 classes.

As a result, when we compared the 0.25 accuracy of the model using Sigmoid as its activation function with the 0.54 accuracy of the model with 5 convolutional layers which used ReLU as the activation function and the Xavier initialization method, we can see that we can expect a large range of performance improvements from parameter tuning. Also, we can expect positive improvements in results when tuning is done in the same fashion even when classes are added.

We can increase the number of classes in the models that we implemented in this study, and we can expand them into systems which can distinguish brand logos and similar designs. In addition, we can develop these models into technology which examines the images included in user feeds and detects trends in fashion, culture, etc.

ACKNOWLEDGEMENT

This research was supported by a grant [KCG-01-2017-05] through the Disaster and Safety Management Institute funded by Korea Coast Guard of Korean government, and it was also supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2015R1D1A1A01060105).

REFERENCES

[1] C. Huang, M. Sushkov, “InstaNet: Object Classification Applied to Instagram Image Streams”, Stanford Computer Science, 2016.

[2] H. Kagaya, K. Aizawa, "Highly Accurate Food/Non-Food Image Classification Based on a Deep Convolutional Neural Network", International Conference on Image Analysis and Processing, pp. 350-357, 2015.

[3] A. Singla, L. Yuan, T. Ebrahami, “Food/Non-food Image Classification and Food Categorization using Pre-Trained GoogLeNet Model”, Multimedia Assisted Dietary Management, pp. 3-11, 2016.

[4] S. Karayev, M. Trentacoste, H. Han, A. Agarwala, T. Darrell, A. Hertzmann, H. Winnemoeller, “Recognizing Image Style”, arXiv : 1311.3715, 2013.

[5] T. Lin, A. RoyChowdhury, S. Maji, “Bilinear CNN Models for Fine-Grained Visual Recognition”, International Conference on Computer Vision, pp. 1449-1457, 2015

[6] Y. Kim, "Convolutional Neural Networks for Sentence Classification", Conference on Empirical Methods on Natural Language Processing, 2014

[7] R. Girshick, J. Donahue, T. Darrell, J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”, Conference on Computer Vision and Pattern Recognition, pp. 580-587, 2014

[8] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, “Return of the Devil in the Details: Delving Deep

into Convolutional Nets", The British Machine Vision Conference, 2014

[9] L. Xu, J. S. Ren, C. Liu, J. Jia, "Deep Convolutional Neural Network for Image Deconvolution", Neural Information Processing Systems, pp. 1790-1798, 2014

[10] R. Girshick, "Fast R-CNN", International Conference on Computer Vision, pp. 1440-1448, 2015

[11] X. Glorot, Y. Bengio, "Understanding the Difficulty of Training Deep Feedforward Neural Networks", Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249-256, 2010.