Journal of information and
communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 16(3): 160-165, Sep. 2018 Regular paper

Convolutional Neural Network Based Image Processing System

Hankil Kim[1], **Jinyoung Kim**[2], **and Hoekyung Jung**[2*], ***Member*, *KIICE***

[1]Department of Music & Sound Technology, Korea University of Media Arts, Sejong 30056, Korea
[2]Department of Computer Engineering, Pai Chai University, Daejeon 35345, Korea

Abstract

This paper designed and developed the image processing system of integrating feature extraction and matching by using convolutional neural network (CNN), rather than relying on the simple method of processing feature extraction and matching separately in the image processing of conventional image recognition system. To implement it, the proposed system enables CNN to operate and analyze the performance of conventional image processing system. This system extracts the features of an image using CNN and then learns them by the neural network. The proposed system showed 84% accuracy of recognition. The proposed system is a model of recognizing learned images by deep learning. Therefore, it can run in batch and work easily under any platform (including embedded platform) that can read all kinds of files anytime. Also, it does not require the implementing of feature extraction algorithm and matching algorithm therefore it can save time and it is efficient. As a result, it can be widely used as an image recognition program.

Index Terms: Artificial intelligence, Convolutional neural network (CNN), Deep learning, Pattern recognition

I. INTRODUCTION

The basic image classification system can classify object types of images by using image features. However, this system can classifies the image types simply by using the characteristics and obtain the result of the inaccurate classification. To solve this drawback, a classification system is designed using a machine learning algorithm [1]. The machine learning algorithms includes k-nearest neighbor, support vector machine (SVM) and artificial neural network [2].

Companies such as Google have been very interested such as deep learning of artificial intelligence and neural networks due to the use of deep neural network to image classification, efforts to efficiently classify images using the deep learning method have been actively pursued [3-5]. There are models such as convolutional neural network (CNN), restricted Boltzmann machine (RBM), and deep belief network (DBN) depending on the application of deep learning [6-12].

Conventional image recognition systems have disadvantages in that feature extraction and matching are handled independently of each other. In this paper, we propose an image processing system that combines feature extraction and matching using CNN technology.

In this paper, we research about deep image classification technology using CNN, and design and implement image processing system based on it. Deep image recognition has also been designed and developed to improve on the low accuracy situations of traditional image classification methods. In this paper, an image processing system is constructed by feature extraction and matched integration through the CNN technology. The algorithm of stochastic gradient descent method and normalization of weights are studied and implemented.

Received 18 May 2018, **Revised** 13 July 2018, **Accepted** 16 July 2018
***Corresponding Author** Hoekyung Jung (E-mail: hkjung@pcu.ac.kr, Tel: +82-42-520-5640)
Department of Computer Engineering, Pai Chai University, 155-40 Baejae-ro, Seo-gu, Daejeon 35345, Korea.

Open Access **https://doi.org/10.6109/jicce.2018.16.3.160** print ISSN: 2234-8255 online ISSN: 2234-8883

The composition of this paper is as follows: Section II describes the design of image processing system based on CNN which is the main focus of this paper. Sections III and IV describe the implementation and performance evaluation of the proposed system. Finally, in Section V, conclusion and future papers of the proposed system are described.

II. SYSTEM DESIGN

In this section, we have designed an image processing system based on a CNN. The system designed in this paper consists of four steps as shown in Fig. 1. In the first step, we design a system of CNN based learning models. In the second step, the detailed calculation process is designed. In the third step, the back propagation/gradient descent algorithm is designed. The final step is to design the calculation optimization algorithm for the system. In particular, the optimal parameters of the learning model are calculated through the second and third steps.

The model is designed to obtain the optimal effect by taking advantage of the existing CNN learning model. To construct this objective, the proposed model has been designed as follows. The convolutional layer can extract feature values for input data using a small size kernel. The activation layer performs nonlinear transformation calculations on input values using nonlinear functions. The pooling layer reduces the dimensionality of the data and can eliminate noise in the data. The inception layer can extract various feature values of data by using two or more convolutional layers in parallel. The skip connection layer is designed using inception layer and convolutional layer to overcome and suppress the feature loss problem of data according to the model dipping

The proposed composite structure model has a group with an inception layer and a skip connection layer as shown in Fig. 2. The function of the inception layer of the model processes the input data using various size kernels at the same time, and outputs the result of parallel connection of various results. The function of the skip link layer can reduce the calculation amount and increase the calculation speed according to the model dip. This can be done by modifying the functions of the skip connection layer and the inception layer when modifying the structure of the proposed model according to the purpose of the learning.

The detailed calculation process includes the following parts. The computation of the convolutional layer is designed by using the filter operation method, and the computation of the pooling layer is designed by using the maximum pooling and the average polling method. The computation of the active layer is performed by using a rectified linear unit (ReLU) nonlinear function to perform the conversion operation on the output values of the convolution product layer.

In order to design the calculation of the inception layer, the following three steps are included. The first step is to output the processing result of the input data by using various size filters (two or more kinds) of the convolutional layer and pooling layer. In the second step, the result of the first step is subjected to the concatenation operation using the CONCAT function. In the third step, the result of the second step is subjected to the dimensional zoom transformation.

In Fig. 3, the skip connection involves a three step operation. In the first step, multidimensional data is output for the dimension of the input data using the convolutional/pulling layer and the inception layer. In the second step, the conversion operation is performed on the number of dimensions of the input data. The third step calculates the result of the addition using the Add function for the first output and the second output. The advantage of this structure is that it can extract, assemble, and optimally learn various features of data through three methods: dip, width, and skip.

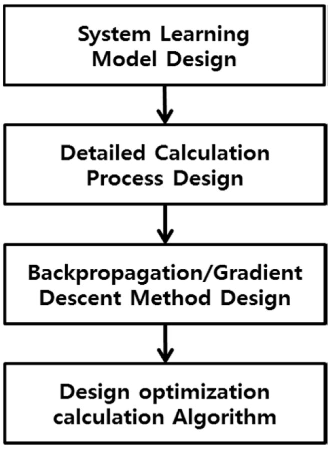

System Learning Model Design
Detailed Calculation Process Design
Backpropagation/Gradient Descent Method Design
Design optimization calculation Algorithm

Fig. 1. System architecture.

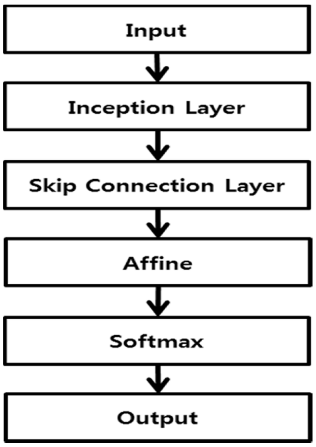


Fig. 2. Composite structure model.

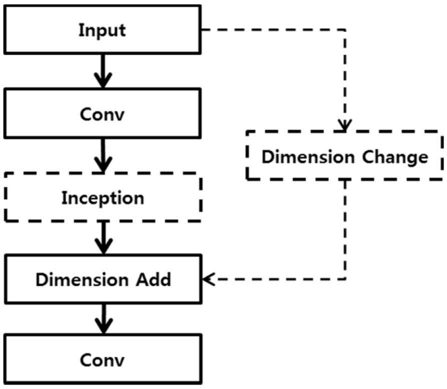


Fig. 3. Skip connection operation.

In order to calculate the correction value of each layer parameter of model using the error between the predicted value and the target value in the back propagation calculation, the following steps are designed. In the first step, the partial differential value of the error between the predicted value and the target value is calculated using (1). In the second step, the partial differential value is calculated for the last hidden layer using (2). In the final step, the partial differential value is calculated using (3) for each hidden layer of the learning model. The gradient descent updates each parameter of the system using (4) and the value in the opposite direction.

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{Y}} = \overline{\Sigma_1^m (\boldsymbol{Y}_m - \boldsymbol{T}_m)^2} \tag{1}$$

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{W}_n} = \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{Y}} \cdot \frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{W}_n} \tag{2}$$

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{W}} = \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{Y}} \cdot \frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{W}_n} \cdot \frac{\partial \boldsymbol{W}_n}{\partial \boldsymbol{W}_{n-1}} \cdots \frac{\partial \boldsymbol{W}_2}{\partial \boldsymbol{W}_1} \tag{3}$$

$$\mathbf{W}_{\mathbf{new}} = \mathbf{W}_{\mathbf{old}} - \eta\theta \tag{4}$$

L: loss value,
Y: output value,
m: prediction value,
T: truth value,
W: weight value,
n: hierarchy,
η: learning rate,
θ: value of the reverse calculation.

The calculation optimization algorithm includes the following parts. The parameter initial value is designed based on the initial value algorithm (5) to suppress the parameter loss problem. The mini-batch normalization is designed based on the normalization algorithm (6) to suppress the overfitting problem. The optimization gradient descent is designed by using modified values and momentum of the learning rate based on the Adam algorithm (7) in order to improve the inefficient calculation problem of the gradient descent.

$$stddev = sqrt\left(\frac{2}{dpper\_size}\right) \tag{5}$$

$$\begin{cases} M = \frac{1}{n}\Sigma_{i=1}^{n} x_i \\ D = \frac{1}{n}\Sigma_{i=1}^{n}(x_i - M)^2 \\ X_i = \frac{x_i - M}{\sqrt{D+K}} \end{cases} \tag{6}$$

$$\begin{cases} M \leftarrow \alpha M - \eta\frac{\partial L}{\partial W} \\ V \leftarrow V + \frac{\partial L}{\partial W} \odot \frac{\partial L}{\partial W} + K \\ W \leftarrow W - \eta\frac{M}{\sqrt[2]{V}}\frac{\partial L}{\partial W} \end{cases} \tag{7}$$

A: input value,
D: dispersion value,
M: mean value,
X: step value,
α: learning coefficient,
K: small value,
V: verification value.

III. SYSTEM IMPLEMENTATION

This chapter deals with the implementation and test results of the image processing system based on the CNN.

The environment for implementing the system is python language, OPENCV, KERAS, CUDA, and NUMPY programs. It is implemented in platforms such as Windows 10 and Linux. The system performance verification is Windows 10 (Inter Pentium CUP 3.20 GHz, 4.00 GB; 2.3 GHz, 2 GB) operating system.

In Fig. 4, the system shows the image processing results for the image (248 × 300) using the convolution layer, the polling layer, and the active layer of various size filters.

Fig. 5 shows the processing results of the image according to the dipping of the model using the optimization calculation algorithm, and compares the processing results of Fig. 6 (not using the optimization calculation). When using the optimization calculation, fitting problems were suppressed.

Fig. 7 shows the learning and test implementation of the image processing system. The system calculates optimal parameters using 80% of the data (training data) and shows the objective accuracy of the system through 20% of the data (test data).

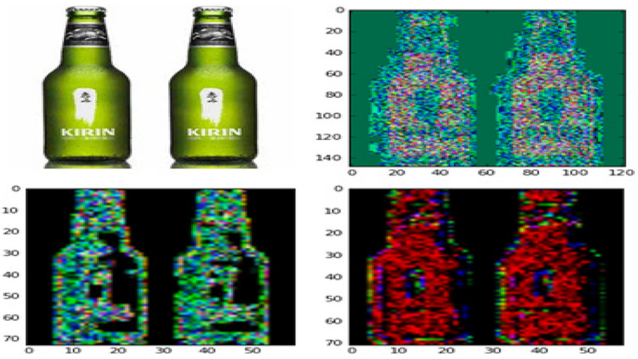


Fig. 4. Processing result by system convolution/polling/active layer.

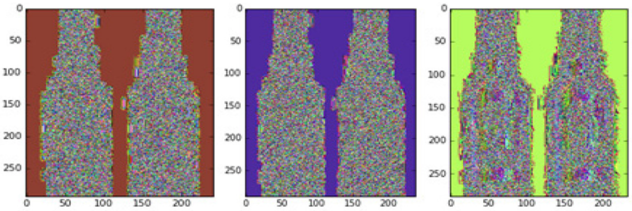


Fig. 5. Processing result by system optimization calculation.

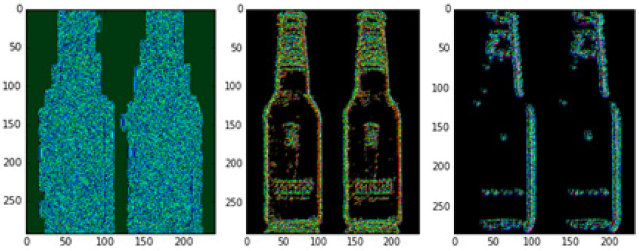


Fig. 6. Processing result by system non-optimization calculation.

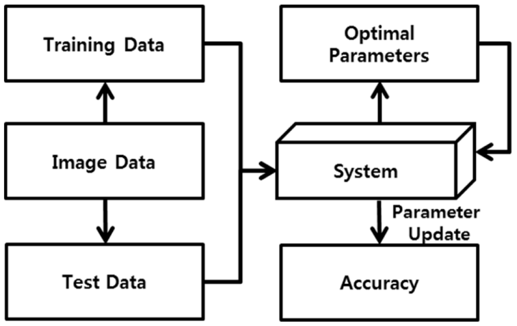


Fig. 7. Implementing the learning and testing process of the system.

IV. REVIEW

Table 1 shows the performance comparison between the proposed image processing system and the basic image recognition system. Test methodology is setting the convolution size and parameter amount of each model. And, we records accuracy and learning time by time.

In Fig. 8, the proposed system uses data to learning and test. Fig. 8 shows the proposed system accuracy according to epoch. Solid lines (the accuracy of the model's test process) indicate better accuracy than the dotted line (the accuracy of the model's learning process). The proposed system accuracy was 84.78%-92.41%.

According to the test results, the proposed system proved to be higher than the 36% accuracy of the basic SVM model system for the 10 images classification processing. The SVM is machine learning algorithm for pattern recognition and data analysis. And, the basic SVM model is general model of image processing systems.

The proposed system shows the test results on the PC and android platforms in Figs. 9 and 10.

V. CONCLUSION

In this paper, we study the deep learning technology and design and implement a convolution based image processing system. The proposed system has 84.78%-92.41% accuracy

Table 1. Performance comparison of basic and proposed system

System type	Detailed structure	Accuracy (%)
Proposed system	Skip connection, Inception	84.78-92.41
Mostly existing systems	SVM	36

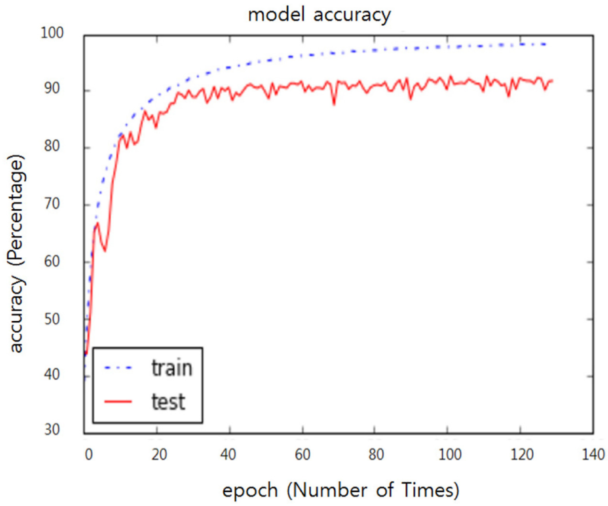
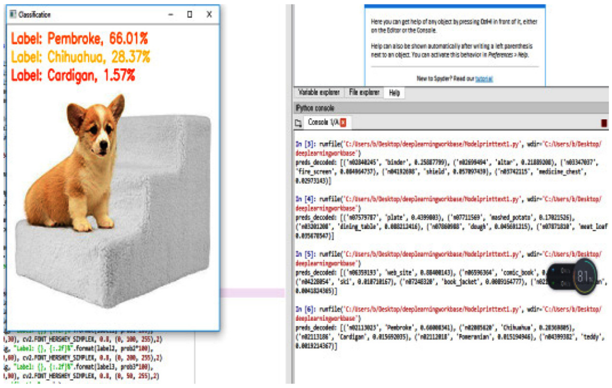


Fig. 8. Accuracy of proposed system.



```
Label: Pembroke, 66.01%
Label: Chihuahua, 28.37%
Label: Cardigan, 1.57%
```

```
,30), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 100, 255),2)
g, "Label: {}, {:.2f}%".format(label2, prob2*100),
,60), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 200, 255),2)
g, "Label: {}, {:.2f}%".format(label3, prob3*100),
,90), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 50, 255),2)
```

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our tutorial

```
In [3]: runfile('C:/Users/b/Desktop/deeplearningworkbase/Modelprinttext1.py', wdir='C:/Users/b/Desktop/
deeplearningworkbase')
preds_decoded: [[('n02840245', 'binder', 0.25887799), ('n02699494', 'altar', 0.21889208), ('n03347037',
'fire_screen', 0.084064737), ('n04192698', 'shield', 0.057097439), ('n03742115', 'medicine_chest',
0.02973143)]]

In [4]: runfile('C:/Users/b/Desktop/deeplearningworkbase/Modelprinttext1.py', wdir='C:/Users/b/Desktop/
deeplearningworkbase')
preds_decoded: [[('n07579787', 'plate', 0.4399083), ('n07711569', 'mashed_potato', 0.17821526),
('n03201208', 'dining_table', 0.088213416), ('n07860988', 'dough', 0.045681215), ('n07871810', 'meat_loaf
0.035678547)]]

In [5]: runfile('C:/Users/b/Desktop/deeplearningworkbase/Modelprinttext1.py', wdir='C:/Users/b/Desktop/
deeplearningworkbase')
preds_decoded: [[('n06359193', 'web_site', 0.08400143), ('n06596364', 'comic_book', 0.[illegible]
('n04228054', 'ski', 0.018710167), ('n07248320', 'book_jacket', 0.0089154777), ('n02[illegible]an',
0.0041824305)]]

In [6]: runfile('C:/Users/b/Desktop/deeplearningworkbase/Modelprinttext1.py', wdir='C:/Users/b/Desktop/
deeplearningworkbase')
preds_decoded: [[('n02113023', 'Pembroke', 0.66008341), ('n02085620', 'Chihuahua', 0.28369805),
('n02113186', 'Cardigan', 0.015692035), ('n02112018', 'Pomeranian', 0.015194946), ('n04399382', 'teddy',
0.0019214367)]]
```

Fig. 9. Test results of the proposed system (PC platform).

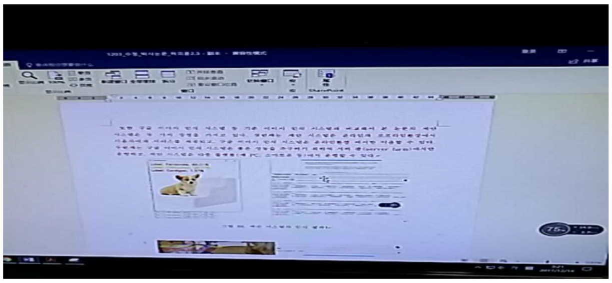


Fig. 10. Test results of the proposed system (Android platform).

to 10 images classification processing and can be implemented in multiple platforms. In addition, overfitting and parameter loss of the basic system are suppressed by using rules and initialization methods.

The proposed system has the following advantages and functions. The accuracy of the proposed system (about 84%–92.41%) increases more than twice the accuracy of the basic system (about 36%) and can be set and supported on multiple-platforms. The proposed system suppresses the problems such as parameter/hardness loss, overfitting of the deep CNN through a specific algorithm (initial value setting, normalization) and pursues optimal accuracy.

Future research should improve the classification performance of the proposed system for more than 100 kinds of classification problems, and second, it should implement the function of searching the position of objects in images.

ACKNOWLEDGMENTS

This study was supported by the research grant of Pai Chai University in 2018.

REFERENCES

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al., "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[2] M. Henrion, "Propagating uncertainty in Bayesian networks by probabilistic logic sampling," *Machine Intelligence and Pattern Recognition*, vol. 5, pp. 149-163, 1988. DOI: 10.1016/B978-0-444-70396-5.50019-4.

[3] C. Ma, C. Chen, Q. Liu, H. Gao, Q. Li, H. Gao, and Y. Shen, "Sound quality evaluation of the interior noise of pure electric vehicle based on neural network model," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 12, pp. 9442-9450, 2017. DOI: 10.1109/TIE.2017.2711554.

[4] L. C. Jiao, S. Y. Yang, F. Liu, S. G. Wang, and Z. X. Feng, "Seventy years beyond neural networks: retrospect and prospect," *Chinese Journal of Computers*, vol. 39, no. 8, pp. 1697-1716, 2016.

[5] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Deep, big, simple neural nets for handwritten digit recognition," *Neural Computation*, vol. 22, no. 12, pp. 3207-3220, 2010. DOI: 10.1162/NECO_a_00052.

[6] J. Schmidhuber, "Deep learning in neural networks: an overview," *Neural Networks*, vol. 61, pp. 85-117, 2015. DOI: 10.1016/j.neunet.2014.09.003.

[7] J. Y. Kim, S. W. Heo, and J. T. Lim, "A license plate recognition algorithm using multi-stage neural network for automobile black-box image," *Journal of the Korea Information and Communication Engineering*, vol. 22, no. 1, pp. 40-48, 2018.

[8] P. V. Kumaraguru and V. J. Chakravarthy, "An image feature extraction and image representation strategies for the analysis of image processing," *Indian Journal of Forensic Medicine & Toxicology*, vol. 11, no. 12, pp. 642-648, 2017. DOI: 10.5958/0973-9130.2017.00202.X.

[9] W. K. Wong, Z. Lai, J. Wen, X. Fang, and Y. Lu, "Low-rank embedding for robust image feature extraction," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2905-2917, 2017. DOI: 10.1109/TIP.2017.2691543.

[10] M. I. Jordan and T. M. Mitchell, "Machine learning: trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255-260, 2015. DOI: 10.1126/science.aaa8415.

[11] A. Hayashi, K. Ishizaki, G. Koblents, and V. Sarkar, "Machine-learning-based performance heuristics for runtime CPU/GPU selection," in *Proceedings of the Principles and Practices of Programming on the Java Platform*, Melbourne, FL, pp. 27-36, 2015. DOI: 10.1145/2807426.2807429.

[12] K. Shin, C. Hwang, and H. Jung, "NoSQL database design using UML conceptual data model based on Peter Chen's frame- work," *International Journal of Applied Engineering Research*, vol. 12, no. 5, pp. 632-636, 2017.

Hankil Kim

received the B.S. and M.S. degrees from the Department of Electronic Engineering of Hanbat National University, Korea, in 2006 and 2011, respectively and the Ph.D. degree in 2015 from the Department of Computer Engineering of Pai Chai University, Korea. Since 2006, he has worked in the Department of Music & Sound Technology at Korea University of Media Arts as a professor. His current research interests include multimedia sound, broadcast sound, sound reinforcement (SR), sound engineering, and database.

Jinyoung Kim

received the B.S. degree in 2001 and M.S. degree in 2011 from the Department of Information Technology of Chonbuk National University, Korea. Since 2001, he has worked for K-Water as a manager especially in Water Treatment Plants. He is currently in a Doctorate course in Department of Computer Engineering of Pai Chai University. His current research interests include 5G, cloud computing, machine learning, bigdata, and artificial intelligence.

Hoekyung Jung

received the M.S. degree in 1987 and Ph.D. degree in 1993 from the Department of Computer Engineering of Kwangwoon University, Korea. From 1994 to 2005, he worked for ETRI as a researcher. Since 1994, he has worked in the Department of Computer Engineering at Pai Chai University, where he now works as a professor. His current research interests include multimedia document architecture modeling, machine learning, IoT, bigdata, and artificial intelligence.