JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY
https://e-journal.uum.edu.my/index.php/jict

How to cite this article:
Shin, S. Y., Jo, G., & Wang, G. (2023). A Novel Method for Fashion Clothing Image Classification Based on Deep Learning. *Journal of Information and Communication Technology, 22*(1), 127-148. https://doi.org/10.32890/jict2023.22.1.6

A Novel Method for Fashion Clothing Image Classification Based on Deep Learning

[1]Seong-Yoon Shin, *[2]Gwanghyun Jo & [3]Guangxing Wang
[1]School of Software,
Kunsan National University, South Korea
[2]Department Mathematics,
Kunsan National University, South Korea
[3]School of Computer and Big Data Science,
Jiujiang University, China

*gwanghyun, s3397220@kunsan.ac.kr
wanggxrs@163.com
*Corresponding author

Received: 8/8/2022 Revised: 19/9/2022 Accepted: 26/9/2022 Published: 19/1/2023

ABSTRACT

Image recognition and classification is a significant research topic in computational vision and widely used computer technology. The methods often used in image classification and recognition tasks are based on deep learning, like Convolutional Neural Networks (CNNs), LeNet, and Long Short-Term Memory networks (LSTM). Unfortunately, the classification accuracy of these methods is unsatisfactory. In recent years, using large-scale deep learning networks to achieve image recognition and classification can improve classification accuracy, such as VGG16 and Residual

Network (ResNet). However, due to the deep network hierarchy and complex parameter settings, these models take more time in the training phase, especially when the sample number is small, which can easily lead to overfitting. This paper suggested a deep learning-based image classification technique based on a CNN model and improved convolutional and pooling layers. Furthermore, the study adopted the approximate dynamic learning rate update algorithm in the model training to realize the learning rate's self-adaptation, ensure the model's rapid convergence, and shorten the training time. Using the proposed model, an experiment was conducted on the Fashion-MNIST dataset, taking 6,000 images as the training dataset and 1,000 images as the testing dataset. In actual experiments, the classification accuracy of the suggested method was 93 percent, 4.6 percent higher than that of the basic CNN model. Simultaneously, the study compared the influence of the batch size of model training on classification accuracy. Experimental outcomes showed this model is very generalized in fashion clothing image classification tasks.

Keywords: Machine learning, deep learning, computational vision, Convolutional Neural Network (CNN), fashion clothing image classification.

INTRODUCTION

With the fast development of computational technology, studies on artificial intelligence (AI) technology made significant strides in the last few years. Among them, the research and application of artificial intelligence algorithms represented by machine learning have rapidly developed (Hinton & Salakhutdinov, 2006). Machine learning is increasingly used in production and life scenarios. For example, the well-known AI robot AlphaGo has become a professional Go player. It defeated the famous human professional Go player with a big score in 2016 (Silver et al., 2016; Fu, 2016). IBM's Watson intelligent question-answering system, Microsoft's online question-answering system named Microsoft XiaoIce (Zhou et al., 2018), and the combination of AI and the Internet of Things (IoT) have produced exceptional medical care, smart city, intelligent logistics, and smart homes. These are the real scenarios of AI algorithm research and application. The application of AI to fashion clothing is an exciting

research field. For example, they are using deep learning (DL) to perceive and detect fashion clothing materials, clothing image recognition, and classification (Thewsuwan & Horio, 2018; Li et al., 2019). Fashion clothing has always been essential to people's pursuit of life taste. With the rise and rapid development of e-commerce, the network is full of images of fashion clothing. How to find the most trendy and innovative clothing in these images is exactly the problem that AI needs to solve.

Regarding the research on fashion clothing image classification, Thewsuwan and Horio (2018) proposed the realization of clothing classification based on texture features utilizing graph-based representation. The experiment was performed by constructing texture and clothing feature recognizers based on three datasets, which included Brodatz texture, UIUC standard texture image database, and the clothing image dataset. Li et al. (2019) proposed a multi-weighted Convolutional Neural Network (CNN) method to realize the retrieval of real-world clothing images. In this research, Li et al. (2019) took the Alex CNN (Krizhensky et al., 2012) large-scale neural network as the basic model. They set up multiple weights to complete the search and classification of the clothing image dataset. Compared with CNN, LeNet, Long Short-Term Memory network (LSTM), and Bidirectional LSTM (BiLSTM) models, the experimental results of these two methods had certain advantages. However, the classification accuracy of Thewsuwan and Horio's (2018) work is still not high enough, at only almost 88 percent. The method of Li et al. (2019) is an experiment based on a large-scale image network. The experimental dataset had millions of images, and the processing was time-consuming.

To overcome this problem, the current study chose CNN as the basic model. Unlike Li et al. (2019), this study adopted a dynamic learning rate decay algorithm and used Rectified Linear Unit (ReLU) as the activation function to optimize the output of each layer in the method. During the training process, several strategies were used to restrain the overfitting of training outcomes, such as dataset partitioning, data augmentation (Wolfe & Lundgaard, 2019), and dropout (Diaz et al., 2017; Gal & Ghahramani, 2015), and finally, the purpose of improving the accuracy of the model was achieved. This paper utilized the Fashion-MNIST dataset as the experimental dataset (Xiao et al., 2017) and compared it with the classification results of cutting-edge

models, such as CNN, LeNet, LSTM, and BiLSTM. The experimental results indicated that the enhanced model had a small structure depth, visible feature extraction effect, and excellent classification accuracy. There are many CNN applications in Korea. Nevertheless, looking at recent research, Park and Choi (2020) presented successful data collection and CNN design for categorizing fish species in inland waters. Eom and Bang (2021) performed Speech Emotion Recognition (SER) using a two-dimensional CNN (2D-CNN) with the Mel-Frequency Cepstrum coefficient.

RELATED WORKS

DL models have been widely used in image recognition and classification for a few years. There are many methods for improving and optimizing DL models. For instance, Gambella et al. (2019) proposed optimization methods based on data augmentation and dropout parameter adjustment, which are very helpful for feature extraction and fine-grained analysis of input data. Ghods and Cook (2019) proposed a model optimization method based on the parameter adjustment of the hybrid model, which mainly started from the optimization of the activation function and the classification function, and further optimized the classification function by selecting the appropriate activation function to improve the prediction accuracy of the model.

Ranjbar et al. (2012) suggested an optimization method based on a loss function (e.g., minimizing a loss function), i.e., to optimize the model by adjusting the parameters of the model (e.g., changing the weight of the model). However, Bengio (2013) pointed out that local minima and discomfort might further complicate DNN optimization in training. Moreover, there are many methods to achieve model optimization by changing the model structure or intensifying the model depth. Large-scale neural network models, for instance, AlexNet, VGG, GoogLeNet, and ResNet are variants of DL models (Krizhensky et al., 2012; Simonyan & Zisserman, 2014; Szegedy et al., 2015; He et al., 2016). These are usually used on large-scale datasets to train models for processing complex image or video problems, such as image recognition and classification, semantic segmentation, and video optimization.

According to the above survey, it can be concluded that the main optimization methods of DL models focus on preprocessing of input data, optimization of model structure, optimization of feature extraction methods, and optimization of model parameters. This paper focuses mainly on optimizing model structure and learning rate. In addition, various studies (Medina et al., 2022; Ma et al., 2022; Vijayaraj et al., 2022), such as deep learning, recognition, and classification for fashion clothing images are currently being conducted.

METHODOLOGY

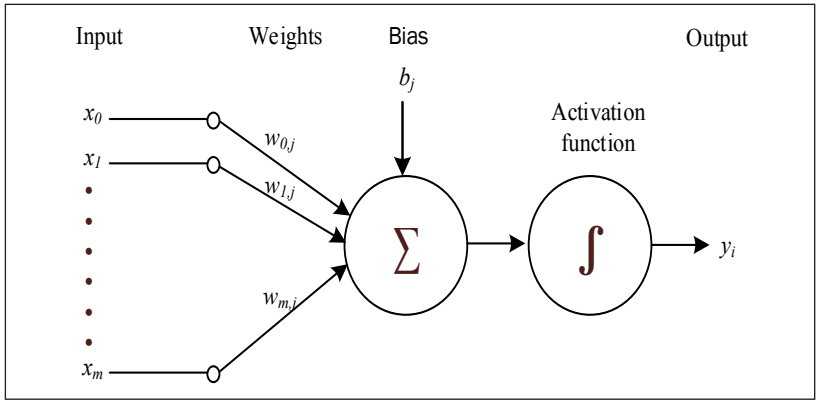
This section briefly introduces the basics of Artificial Neural Networks (ANNs) and CNNs. This is the theoretical basis of the current research work.

Artificial Neural Networks

The DL model in this study was based on an ANN, which is a fully connected perceptron from input to output through the cascade. Figure 1 shows an ANN structure. The concept of artificial neurons was first proposed by McCulloch and Pitts and developed into the concept of perceptron in 1958 (McCulloch & Pitts, 1943; Rosenblatt, 1958). Dodd and McCulloch (1989) proposed a neural network structure based on the Markov process. Each node (perceptron) has a mathematical model that can express the information transfer of specific biological neurons. Generally, ANN consists of a neuron structure with multiple layers. Each layer of neurons has an input and an output. The layers (using the symbol as layer i) are composed of N_i (N_i represents N on the i-th layer) network neurons. The network neurons on each N_{i} correspond to $N_{i\text{-}1}$. The neurons that output from the upper layer are used for the input of the next layer. In neural network models, Sigmoid is usually applied as an activation function to control the output size. Because the Sigmoid function is differentiable and continuous, it can easily handle the delta rule. Finally, the output of the neuron in the ANN model is obtained. It can be seen that the output is non-linear. In other words, the value gained by the excitation function determines whether the neuron is activated according to the threshold value. The neural unit is defined in Equation 1.

Figure 1

The Unit of Artificial Neural Network



$$y_j = f\left(\sum_{i,j=0}^{m-1} w_{i,j}x_i + b_j\right) \quad (1)$$

In Equation 1, $x_0, x_1, ..., x_m$ are the components of the input vector, w_{0j}, $w_{1j}, ..., w_{mj}$ are the weights of each synapse of the neural unit, b_j is the bias term, $f(x)$ is the transfer function that is usually non-linear, and y_j is the neuron output. Equation 1 indicates a mathematical representation of a single layer of neurons. The mathematical expression of a multilayer neural network composed of multiple neurons is shown in Equation 2.

$$y_j^k = f\left(\sum_{i,j=0}^{m-1} w_{i,j}^k x_i^{k-1} + b_j^k\right) \quad (2)$$

Where w^k_{ij} is the weight value of the *j*th output of the *i*th node in the *k*th layer of the *k-1*th layer, b^k_j is the bias term of the *j*th node of the *k*th layer, and function $f(x)$ is a non-linear activation function. The activation function can be seen as a filter that receives diverse external signals and adjusts its function to the outcome of the expected value. ANNs typically use three activation functions: threshold, piecewise, and continuous bipolar (e.g., Sigmoid, Tanh). The total number of nodes of the *k*-1 layer in the multilayer neural network is m_{k-1}.

Convolutional Neural Network (CNN)

Convolutional Neural Network, also referred to as CNN or ConvNet, is widely used in many tasks, such as computational vision and

speech/language recognition. In the 2012 ImageNet Challenge, Krizhensky et al. (2012) achieved impressive results by applying deep CNNs for the first time. Since then, the architectural design of deep CNNs has attracted many researchers to contribute. CNNs usually consist of input layers, convolutional layers, pooling layers, fully connected, and output layers. The input layer is a set of input vectors, i.e., a set of vectors input to the neurons of the neural network. A convolutional layer is a collection of parallel feature function maps, which are feature function maps that consist of sliding differential convolutional kernels on an input image and performing a specific operation. In each sliding location, the convolution kernel and input image perform multiplication/sum operations corresponding to the elements and project the information from the perceptron onto the elements in the functional map. The process of sliding is called stride. Suppose $H \times W \times C$ represents the length, width, and dimension of the input image, Z_s represents the stride, Z_p represents the filling size of the incomplete coverage of the convolutional layer, and $k_1 \times k_2 \times$C indicates the size of the convolution kernel. The mathematical calculation formula for the size of the layer is shown in Equation 3.

$$Dim_c\left(H_1, W_1, D_1\right) = \left(\frac{H + 2Z_{p-k_1}}{Z_s} + 1\right), \left(\frac{W + 2Z_{p-k_2}}{Z_s} + 1\right), K_D \qquad (3)$$

The pooling layer refers to downsampling, which combines the neuron cluster output from the previous layer with a single neuron from the next layer. The work is done after non-linearity. In CNN models, pooling layers can often be used to reduce noise. The commonly used pooling method is the maximum pooling (MaxPooling) method. Average pooling and $L2$ normal form pooling operations can also be used. When the pooling operation is performed with the number of convolution kernels, D_n, and the stride size, Z_s, the mathematical calculation formula of the dimensions of the pooling layer is shown in Equation 4. With Equation 4, a new three-dimensional convolution output unit Dimp (H_2, W_2, D_2) is calculated.

$$Dim_p\left(H_2, W_2, D_2\right) = \left(\frac{H_1 - k}{Z_s} + 1\right), \left(\frac{W_1 - k}{Z_s} + 1\right), D_n \qquad (4)$$

PROPOSED METHOD

This section presents the learning rate approximation method and the detailed structure of the proposed CNN model.

Learning Rate Approximation Method

The learning rate is the ability of neural network training and learning and is the most demanding global parameter to adjust in a neural network. If the learning rate is set too large, the loss rate of model training will oscillate or even not converge. The training process will significantly increase if the learning rate is too low. Generally, the learning rate starts from 0.1 in the neural network model and decreases exponentially, such as 0.01 and 0.001. The learning rate is also named the step size, i.e., η in the back-propagation algorithm (as in Equation 5).

$$\eta : w^n \leftarrow w^n - \eta \frac{\partial L}{\partial W^n} \tag{5}$$

The expression of the learning rate is $lr_t=lr_0/(1+k_t)$, where lr_0 is the initial learning rate, k controls the attenuation range, and t is the number of training rounds. In the CNN model, it is assumed that α is the learning rate, α_0 is the first learning rate, d_r is the hyperparameter of the learning rate decay, and *epoch* is the total number of training iterations. The calculation formula of the learning rate is illustrated in Equation 6.

In the proposed model, the learning rate was tested by an exponential decay. The formula of learning rate exponential decay is shown in Equation 7. According to Equation 7, the first learning rate, α_0, was set and the total number of global iterations, *global_step*, was assigned according to the batch size. Then, the learning rate attenuation parameter, ρ, i.e., the initial attenuation amplitude was initialized. As the iteration progressed, the hyperparameter, d_r, was calculated for each batch. Then, the decay rate of the learning rate was calculated and updated. Finally, the optimal learning rate for this iteration was obtained. Algorithm 1 indicates the proposed method's pseudo-code of the learning rate approximation algorithm.

$$\alpha = \frac{1}{1 + d_r * epoch} * \alpha_0 \tag{6}$$

$$\alpha = \alpha_0 * e^{-d_r * global\_step} \tag{7}$$

Algorithm 1

The Pseudo-Code of an Approximation Algorithm for Dynamic Learning Rate

Algorithm 1: Approximation of Dynamic Rate of Learning

Input:

α_0 = first learning rate

global_step = the total iterations

ρ = rate of decay

d_r = hyperparameter

Output:

Cumulative and computer updates
Learning Rate

1: Cumulative variable initialization $E|g^2|_t = E|\Delta\alpha^2|_{0=0}$

2: **For** t = 1, 2, ..., *global_step* **do**

3: Cumulative Gradient: $E|g^2|_t = \rho E|g^2|_{t-1}+(1-\rho)g_t^2$

4: Computer Update: $\Delta\alpha = RMS|\Delta\alpha|_{t-1}/RMS|g|_t \cdot g_t$

5: Cumulative Update: $E|\Delta\alpha^2|_t = \rho E|\Delta\alpha^2|_{t-1}+(1-\rho)\Delta\alpha_t^2$

6: Computer Learning Rate: $\alpha_t = \alpha_{t-1} * e^{d_r * global\_step}$

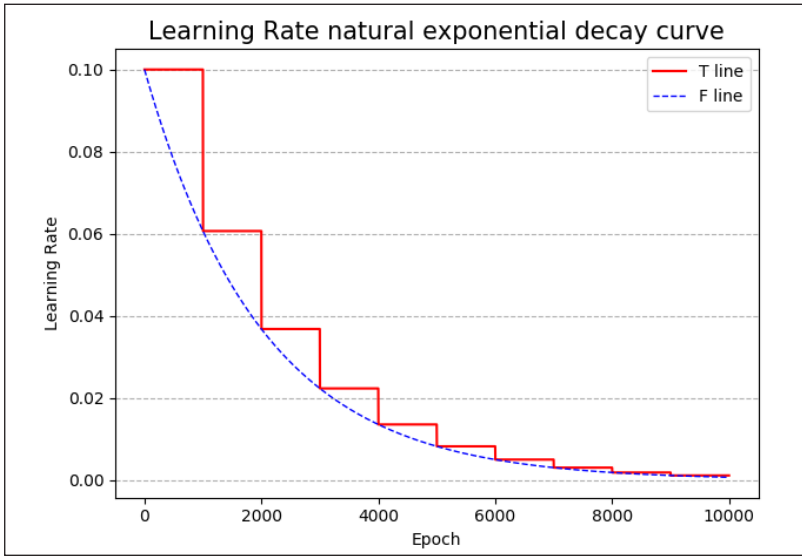
7: Update Learning Rate: $\alpha_{t+1} = \alpha_t + \Delta\alpha_t$

8: **End For**

This study adopted the most suitable learning rate in the model training. The natural exponential learning rate decay curve is presented in Figure 2. In Figure 2, the red line represented the accurate learning rate decay degree, and the blue dotted line indicated the learning rate decay curve. It can be concluded that the learning rate decayed from the initial learning rate of 0.1. After 0.01, the learning rate decayed less until the decay stopped. Therefore, the optimal learning rate should be 0.01. A dynamic learning rate method could accelerate model convergence and prevent overfitting due to the increased decay of the learning rate.

Figure 2

The Curve of the Learning Rate's Natural Exponential Decay



Proposed Model Structure

In this paper, a combination of the convolutional layer and maximum pooling layer was adopted and ReLU was used as the activation function, with the dropout set to 0.25. The proposed model used two combination modules. The first layer size of the convolution kernel was 3×3, and the stride size was 1. The pool size was 2×2 a with a stride size of 1. The stride size of the second combination module was 2. Then, two connected convolutional layers were adopted, and the dropout was set to 0.5. The next layer was the fully connected layer, with the dropout set to 0.5, and Softmax as the classification function. The final layer was the output layer, which produced the classification results. The detailed structural parameters of the proposed model are reported in Table 1. Such a structure realized rapid extraction of input image features, achieved fine-grained sorting and extraction of features through pooling and dropout, and realized classification at the fully connected layer. It could improve the classification accuracy of the model and achieve the optimization purpose. The model's optimizer used an adaptive gradient descent algorithm, such as Adam, to accomplish the rapid convergence of the model. Figure 3 shows the proposed model structure.

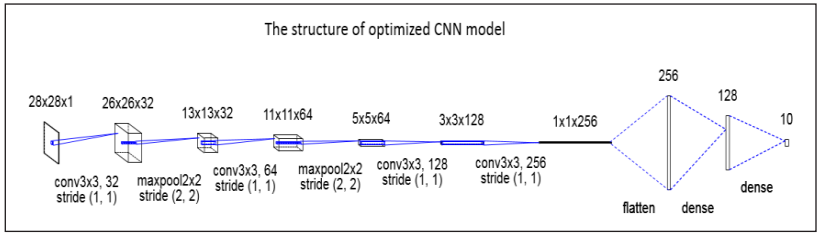
Table 1

Parameters of the Proposed CNN Model

Layers	Type	Convolution Cell	Stride	Output Size	Dropout
I	Input			26×26×1	–
C1	Convolution	3×3	1	26×26×32	–
M1	MaxPooling	2×2	1	13×13×32	0.25
C2	Convolution	3×3	2	11×11×64	–
M2	MaxPooling	2×2	2	5×5×64	0.25
C3	Convolution	3×3	2	3×3×128	–
C4	Convolution	3×3	2	1×1×256	0.5
FC1	Fully Connected			256	–
FC2	Fully Connected			128	0.5
O	Output			10	–

Figure 3

Structure of the Proposed CNN Model



EXPERIMENT AND RESULT ANALYSIS

In this section, the experimental method and results are introduced in detail, and the experimental results are analyzed and discussed.

Implementation Detail

The experimental dataset was from Zalando's article (Zalando Research, 2021). This dataset contained a training set of six million

examples and a testing set of ten thousand examples. Each example was a 28×28 grayscale image associated with ten categories of labels. Moreover, the number of data examples for each class was evenly distributed. In the experiment, divided the training dataset was divided by a ratio of two to eight, divided into 48,000 pieces of training and 12,000 pieces of data for model verification. Each type of data was randomly selected. Then, the model was evaluated on the testing set to obtain prediction results. The training data and validation data were normalized first. In this way, the study could achieve dimensionality reduction and increase the comparability of the data. The size of the preprocessed data was (28, 28, 1). The research team initialized the learning rate to 0.1 and the learning rate range to [0.1, 1e-8]. The learning rate was adjusted dynamically through the learning rate optimization algorithm via Equation 7 and Algorithm 1. The batch size was set at 128, and there were 375 batches in the training data blocks per epoch and 50 training iterations.

The experimental hardware environment was a workstation computer with an Intel Core i7-4790 and a graphics processor NVIDIA GeForce GTX 960 with 2 GB of memory. The computer had 8 GB of RAM and 2 TB of hard drive capacity. This study used the Python programming language to implement the optimization algorithms, build models, and adopt the TensorFlow framework with the Keras DL library as the back-end computing tool.

The experimental dataset was trained with CNN, LeNet, LSTM, BiLSTM, and the proposed model, and evaluated on the same testing set. The study compared the experimental results in prediction accuracy, recall, F1-score, and loss (rate) of the research work. Table 2 shows the results of each model's accuracy and loss rate. Table 3 presents detailed experimental results of the improved model. The accuracy-loss rate curves of the CNN model and the improved model are shown separately in Figures 4 and 5. Figure 6 indicates the confusion matrix for the testing set. Furthermore, different batch sizes were set to observe the accuracy and area under the ROC curve (AUC) trends of the model. The details are shown in Table 4. Figures 7 and 8 show the Receiver-Operating-Characteristic (ROC) and (AUC) (ROC-AUC) curves of the optimization model and the ROC curves of each classification, which directly reflected the prediction and classification performance of the optimization model.

Table 2

Parameters of the Proposed CNN Model

Models	Accuracy (%)	Loss
CNN	89.10	0.37
LeNet	87.65	0.34
LSTM	87.15	0.36
BiLSTM	87.49	0.36
Proposed CNN Model	**92.87**	**0.20**

Table 3

Detailed prediction results of models on the test set (Pre: precision, Rec: recall, F1: f1-score. The number of test samples in each category is 1000)

	CNN			LeNet			LSTM			BiLSTM			Proposed Model		
Category	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
T-shirt/ Top	0.86	0.80	0.83	0.88	0.74	0.80	0.80	0.82	0.81	0.85	0.79	0.82	0.88	0.88	0.88
Trousers	0.98	0.98	0.98	0.99	0.97	0.98	0.99	0.96	0.98	0.99	0.97	0.98	0.99	0.99	0.99
Pullover	0.85	0.77	0.80	0.76	0.86	0.81	0.77	0.80	0.79	0.78	0.79	0.78	0.91	0.86	0.88
Dress	0.92	0.89	0.90	0.89	0.88	0.88	0.82	0.92	0.87	0.87	0.88	0.88	0.93	0.95	0.94
Coat	0.82	0.87	0.84	0.81	0.79	0.80	0.80	0.75	0.77	0.77	0.84	0.80	0.87	0.89	0.88
Sandals	0.96	0.95	0.96	0.99	0.92	0.95	0.98	0.96	0.97	0.97	0.94	0.95	0.99	0.98	0.98
Shirt	0.69	0.78	0.73	0.67	0.70	0.68	0.67	0.62	0.65	0.71	0.66	0.68	0.78	0.80	0.79
Sneakers	0.95	0.93	0.94	0.92	0.96	0.94	0.92	0.98	0.95	0.91	0.96	0.94	0.96	0.97	0.96
Bag	0.96	0.98	0.97	0.96	0.97	0.96	0.97	0.97	0.97	0.95	0.97	0.96	0.99	0.99	0.99
Ankle Boots	0.93	0.97	0.95	0.94	0.97	0.95	0.98	0.95	0.96	0.96	0.95	0.95	0.98	0.97	0.97
avg/ total	0.89	0.89	0.89	0.88	0.88	0.88	0.87	0.87	0.87	0.87	0.87	0.87	**0.93**	**0.93**	**0.93**

Analysis of Experimental Results

The following sub-sections will discuss the experimental results in terms of accuracy, loss rate, confusion matrix, and ROC-AUC curve.

Model Accuracy and Loss Rate

Table 2 showed the assessment accuracy and the loss rate of each model. It can be concluded that the accuracy of the optimized model

was the highest, reaching 92.87 percent, and the loss rate was reduced to the lowest at 0.20. From the table, it can be seen that the proposed method outperformed the other methods.

The Accuracy of the Improved Model

Figures 4 and 5 illustrate that the accuracy of the enhanced model increased remarkably compared with the CNN model. As the number of training iterations increased, the model's validation accuracy gradually kept matching the training accuracy. In addition, Figure 4 indicated that the loss factor (rate) of the CNN model decreased at the beginning of training and then gradually increased, indicating that the model did not converge. In contrast, as shown in Figure 5, the loss rate of the improved model steadily decreased since the beginning of training, and it was less than the loss rate of training. After ten iterations, it stabilized, and the gap between the training and loss rate was negligible. This indicated that the model had a fast convergence rate and high optimization performance.

Figure 4

Accuracy and Loss (Rate) Curves for the Proposed CNN Model

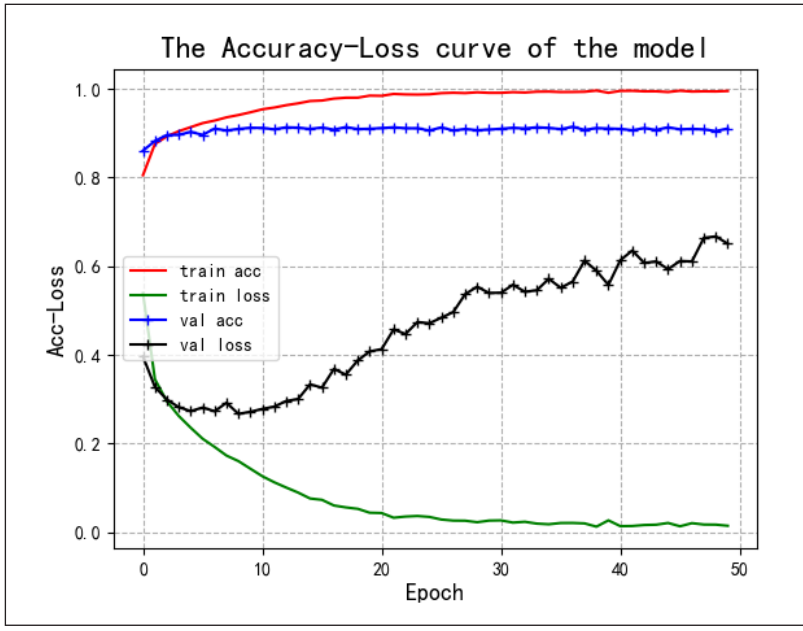
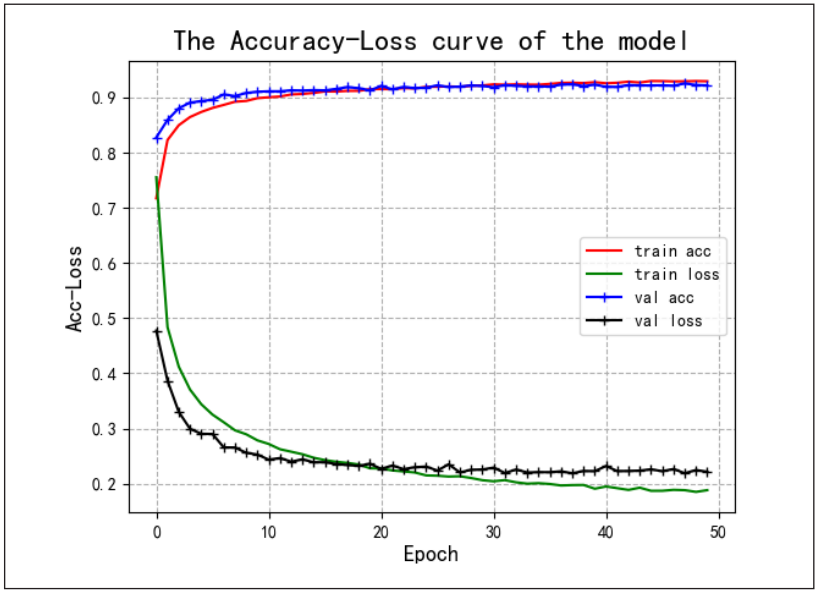


Figure 5

Accuracy and Loss (Rate) Curve for the Presented Model



The details of the prediction accuracy were shown in Table 3, which included evaluation indexes such as precision, recall, and F1-score. Firstly, from the average of each evaluation index, the precision of model prediction was 89 percent for CNN, 88 percent for LeNet, 87 percent for LSTM and BiLSTM, and 93 percent for the proposed method. It indicated that the proposed approach had the best prediction results, followed by the CNN and LeNet models, and the worst results were the LSTM and BiLSTM models. Compared with the result of other models, the accuracy of the proposed model increased by a maximum of 6.9 percent. The experimental results of other indicators were consistent with the evaluation results of prediction accuracy. Secondly, it is noticed that LSTM and BiLSTM were better at dealing with temporal classification tasks. Finally, Table 4 shows the proposed method's accuracy, loss rate, and AUC values under different batch sizes. It can be concluded that with a batch size of 256, the model's accuracy and AUC values were the highest, reaching 92.88 percent and 0.9978, respectively, and the loss rate was the lowest, dropping to 0.1939. The model with a batch size of 128 had better prediction accuracy and AUC values, whereby the accuracy was 92.87 percent,

and the loss rate was 0.2003. In general, batch size had little influence on the proposed method's prediction result.

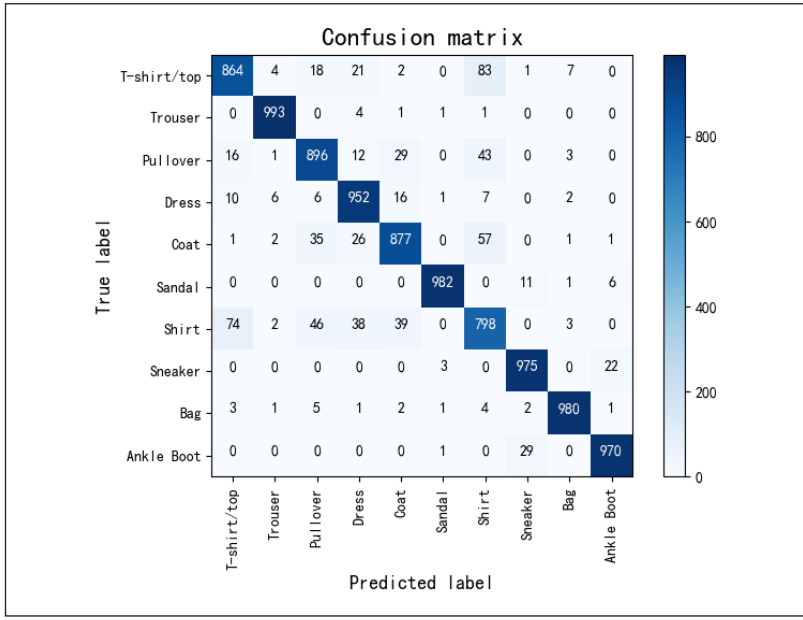
Table 4

Prediction Performance of the Improved Model with Different Batch Sizes

Batch Size	***32***	***64***	***128***	***256***	***512***
Accuracy (%)	*91.70*	*92.42*	*92.87*	***92.88***	*92.69*
Loss	*0.2137*	*0.2034*	*0.2003*	***0.1939***	*0.1997*
AUC	*0.9969*	*0.9976*	*0.9977*	***0.9978***	*0.9976*

Figure 6

Confusion Matrix of the Proposed CNN Model



Confusion Matrix

The confusion matrix of the proposed model is shown in Figure 6, which intuitively reflects the prediction results and classification of the model on the test set. The best predictions were the "Trousers" and "Bag" categories, followed by the "Sandals" category. The worst

result was the "Shirt" category. When this happened, it is estimated that the feature information of the "Shirt" and "Pullover" categories was very similar. Although the improved model had poor prediction results in the "Shirt" category, the results were still better than the other models.

Improved Model's ROC-AUC Curve

The (ROC-AUC) curve of the proposed model is illustrated in Figure 7, which intuitively reflects the classification effect of the optimized model. The AUC value reached 0.998, indicating that the model almost correctly predicted all the images in the testing set. The ROC curve of each category of the optimization model is shown in Figure 8. It can be intuitively concluded from Figure 8 that most classification curves' area of the ROC was very close to 1.00. Only the red curve of class 6 ("Shirt" category) was the lowest, and its AUC value reached 0.98. This result corresponded with the prediction result of the confusion matrix in Figure 6.

Figure 7

ROC-AUC Curve of the Proposed CNN Model (in the ROC-AUC Curve, the Area Under the Red Curve is More Extensive, and the Number that the Model Predicts Samples Correctly is More)

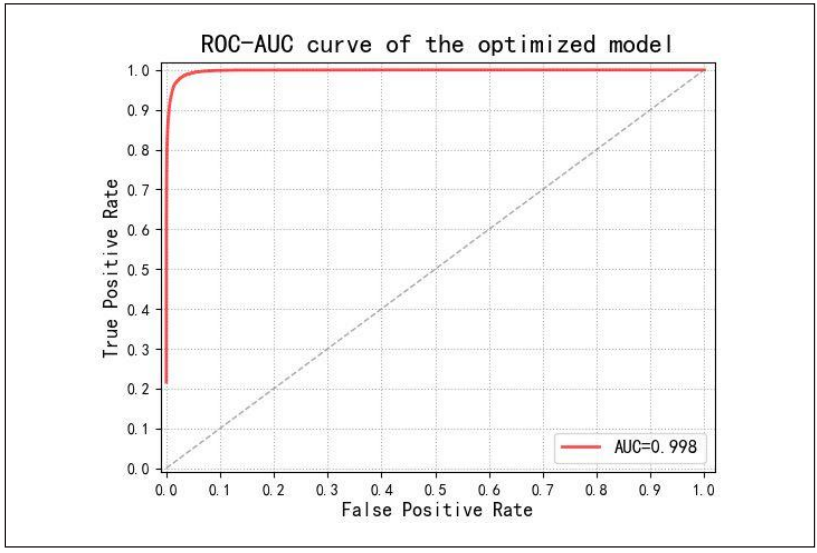
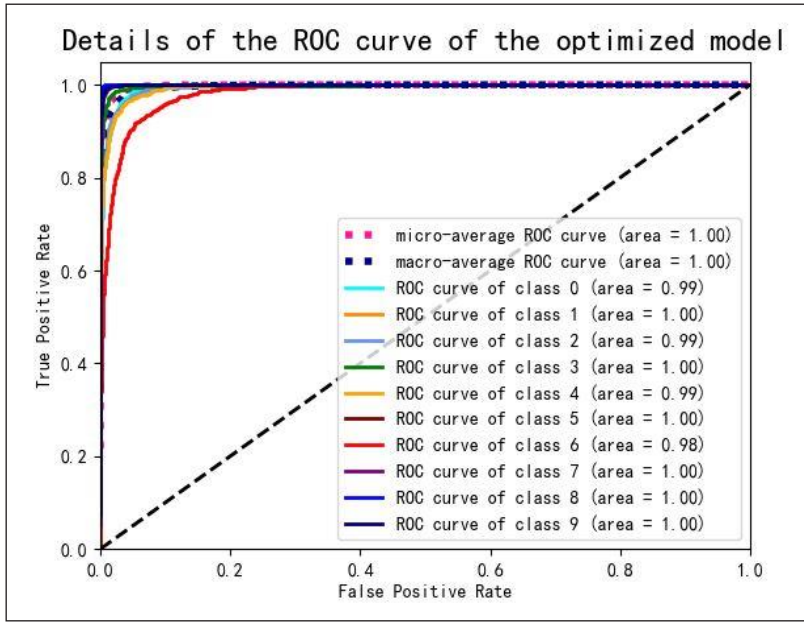


Figure 8

Details of ROC curve of the proposed CNN model (class 0: T-shirt/ Top; class 1: Trousers; class 2: Pullover; class 3: Dress; class 4: Coat; class 5: Sandals; class 6: Shirt; class 7: Sneakers; class 8: Bag; class 9: Ankle Boots).



CONCLUSION

In this study, a novel way was proposed for DL-based fashion clothing image recognition and classification, focusing on optimizing the learning rate and model structure. The study dynamically adjusted the model's learning rate based on the training's batch size. By setting a combination module of convolutional and pooling layers, the proposed model had an outstanding ability to excerpt worthwhile features for the image. The research team set ReLU as the activating function and further increased the number of convolutional layers for granular feature extraction. In the model, the value and position of dropout were rationally set, which not only accelerated the model convergence but also effectively suppressed overfitting. More importantly, Softmax was established as a classification function at the fully connected layer to achieve accurate model classification

and improve classification accuracy. The study performed detailed experiments on the Fashion-MNIST image dataset, and as a result, the accuracy of the proposed model reached 93 percent. Compared with the CNN, LeNet, LSTM, and BiLSTM models, the experimental results of the proposed model had increased accuracy by 4.5 percent, 5.7 percent, and 6.9 percent, respectively. Of course, this research mainly focused on approximating the learning rate and the model structure. The proposed model's parameters also only involved the comparison of the batch size. In the future, this research will focus on how to make the model structure shallower and more effective, reduce training time, and improve image classification accuracy.

ACKNOWLEDGMENT

The second author (Gwanghyun Jo) is financially supported by the National Research Foundation of Korea (NRF) grant, Ministry of Science and ICT (MSIT), South Korea (No. 2020R1C1C1A01005396).

REFERENCES

Bengio, Y. (2013, July). Deep learning of representations: Looking forward. In *SLSP'13 Proceedings of the First International Conference on Statistical Language and Speech Processing* (Vol. 7978, pp. 1–37). Springer. http://dx.doi.org/10.1007/978-3-642-39593-2_1

Diaz, G., Fokoue, A., Nannicini, G., & Samulowitz, H. (2017). An effective algorithm for hyperparameter optimization of neural networks. *IBM Journal of Research and Development*, *61*(4/5), 9–1. https://arxiv.org/pdf/1705.08520

Dodd, N., & McCulloch, N. (1989, October). Structured neural networks for Markovian processes. In *1989 First IEE International Conference on Artificial Neural Networks* (Vol. 313, pp. 319–323). https://ieeexplore.ieee.org/iel3/1151/1872/00051984.pdf

Eom, Y., & Bang, J. (2021). Speech emotion recognition using 2D-CNN with mel-frequency cepstrum coefficients. *Journal of Information and Communication Convergence Engineering*, *19*(3), 148–154. https://doi.org/10.6109/jicce.2021.19.3.148

Fu, M. C. (2016, December). AlphaGo and Monte Carlo tree search: The simulation optimization perspective. *2016 Winter Simulation Conference (WSC)* (pp. 659–670). https://doi.org/10.1109/WSC.2016.7822130

Gal, Y., & Ghahramani, Z. (2015, June). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning* (pp. 1050–1059). *PMLR*. https://arxiv.org/pdf/1506.02142

Gambella, C., Ghaddar, B., & Naoum-Sawaya, J. (2019). Optimization problems for machine learning: A survey. *European Journal of Operational Research*, *290*(3), 807–828. https://arxiv.org/pdf/1901.05331

Ghods, A., & Cook, D. J. (2019). A survey of techniques all classifiers can learn from deep networks: Models, optimizations, and regularization. *Machine Learning*, 1–28. https://arxiv.org/pdf/1909.04791

He, K., Zhang, X., Ren, S., & Sun, J. (2016, June). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). http://dx.doi.org/10.1109/CVPR.2016.90

Hinton, G. E., & Salakhutdinov, R. R. (2006, July). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507. https://doi.org/10.1126/science.1127647

Krizhensky, A., Sutskever, I., & Hinton, G.E. (2012). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Li, R., Feng, F., Ahmad, I., & Wang, W. (2019). Retrieving real world clothing images via multi-weight deep convolutional neural networks. *Cluster Computing*, *22*(3), 7123–7134. https://doi.org/10.1007/s10586-017-1052-8

Ma, W., Tu, X., Luo, B., & Wang, G. (2022). Semantic clustering based deduction learning for image recognition and classification. *Pattern Recognition*, *124*, 108440. https://doi.org/10.1016/j.patcog.2021.108440

McCulloch, W. X., & Pitts, W. (1943) A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, *5*(4), 115–133. https://philpapers.org/go.pl?id=MCCALC-5&proxyId=&u=https%3A%2F%2Fdx.doi.org%2F10.1007%2Fbf02478259

Medina, A., Méndez, J., Ponce, P., Peffer, T., Meier, A., & Molina, A. (2022). Using deep learning in real-time for clothing classification with connected thermostats. *Energies*, *15*(5), 1811. https://doi.org/10.3390/en15051811

Park, J. H., & Choi, Y. K. (2020). Efficient data acquisition and CNN design for fish species classification in inland waters. *Journal of Information and Communication Convergence Engineering*, *18*(2), 106–144. https://doi.org/10.6109/jicce.2020.18.2.106.

Ranjbar, M., Lan, T., Wang, Y., Robinovitch, S. N., Li, Z.-N., & Mori, G. (2012). Optimizing nondecomposable loss functions in structured prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(4), 911–924. https://doi.org/10.1109/TPAMI.2012.168

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. v. d., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., & Hassabis D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–489. https://doi.org/10.1038/nature16961

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 1–14, https://arxiv.org/pdf/1409.1556

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015, June). Going Deeper with Convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9). https://doi.org/10.1109/CVPR.2015.7298594

Thewsuwan, S., & Horio, K. (2018) Texture-based features for clothing classification via graph-based representation. *Journal of Signal Processing*, *22*(6), 299–305. https://doi.org/10.2299/jsp.22.299

Vijayaraj, A., Raj, V., Jebakumar, R., Gururama Senthilvel, P., Kumar, N., Suresh Kumar, R., & Dhanagopal, R. (2022). Deep learning image classification for fashion design. *Wireless Communications and Mobile Computing*, *2022*. https://doi.org/10.1155/2022/7549397

Wolfe C. R., & Lundgaard K. T. (2019). Data augmentation for deep transfer learning. *Machine Learning*, 28, 1–11. http://www.cs.utexas.edu/users/ai-lab/downloadPublication.php?filename=http://nn.cs.utexas.edu/downloads/papers/cwolfethesis.2019.pdf&pubid=127780

Xiao, H., Rasul, K., & Vollgraf, R. (2017) Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *Machine Learning*, 1–6. https://arxiv.org/pdf/1708.07747

Zalando Research. (2021, February). *Fashion MNIST: An MNIST-like dataset of 70,000 28x28 labeled fashion images*. https://www.kaggle.com/datasets/zalando-research/fashionmnist

Zhou, L., Gao, J., Li, D., & Shum, H.-Y. (2018). The design and implementation of XiaoIce, an empathetic social chatbot. *Artificial Intelligence*, 1–35. https://arxiv.org/abs/1812.08989