Sensors and Materials, Vol. 34, No. 9 (2022) 3569–3580
MYU Tokyo

S & M 3061

Malware Classification Using a Taguchi-based Deep Learning Network

Cheng-Jian Lin,[1,2*] Xin-You Lin,[3] and Jyun-Yu Jhang[4]

[1]Department of Computer Science & Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan
[2]College of Intelligence, National Taichung University of Science and Technology, Taichung 404, Taiwan
[3]Ph.D. Program, Prospective Technology of Electrical Engineering and Computer Science, National Chin-Yi University of Technology, Taichung 411, Taiwan
[4]Department of Computer Science & Information Engineering, National Taichung University of Science and Technology, Taichung 404, Taiwan

(Received April 7, 2022; accepted September 2, 2022)

Keywords: malware, convolutional neural network, Taguchi method, orthogonal table

Malware is designed to damage computer systems, and malicious targets have proliferated recently. This rising use of malware requires an efficient malware detection method. Because new malware is constantly being created and old malware is constantly updated, manually updating a signature database with newly generated malware samples is increasingly challenging. To reduce the cost of feature engineering and the requirement for domain expert knowledge, researchers have used image-sensing methods to solve the malware family classification problem. In this study, a Taguchi-based deep learning network (TDLN) with optimization of the parameter combination is proposed for malware family classification. A total of 36 experiments were conducted and nine influential factors with various levels were selected for determining the optimal parameters of the proposed TDLN. The experimental results indicate that the accuracy, precision, and recall of malware family classification when using the proposed TDLN are 98.71, 96.90, and 96.78%, respectively. Moreover, the accuracy, precision, and recall of the proposed TDLN are 2.03, 5.59, and 6.09% higher, respectively, than those of the original deep learning network for the Malimg data set.

1. Introduction

Malicious software is generally disseminated through networks and portable storage devices and causes confidential data leakage, system damage, data loss due to unexpected failures, and information security problems. Because of the rapid development of technology, the use of malware for illegal purposes has increased considerably. For example, the Kaspersky Lab detected 69277289 types of malware in 2016 (including scripts and executable files). McAfee Labs stated in a report that in the past 4 years, the number of malicious software attacks has increased by 22% and has reached 670 million. Despite the rapid increase in malicious software,

*Corresponding author: e-mail: cjlin@ncut.edu.tw
https://doi.org/10.18494/SAM4044

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

limited methods are available for effectively detecting it. Commercial antivirus applications use signature-based analysis; the disadvantage of this method is that the signature database must be updated regularly to detect changing threats. If no method is available for correctly scanning different types of malware viruses, protecting crucial data and resources in real time is impossible.(1)

Malware was previously detected using static or dynamic signature-based techniques.(1) The main disadvantage of traditional signature-based detection methods is that they are not scalable and their effectiveness is undermined as new variants of malware emerge.(2) Many researchers have adopted machine learning to overcome the problems associated with signature-based detection.

Machine learning technology has been widely used to address the malware classification problem. Work on malware classification through machine learning can be broadly classified into two categories: binary- and image-sensing-based methods. Of the binary-based methods, an optimal K-nearest neighbor (KNN) algorithm was proposed by Assegie(3) for malware detection and classification. Chiramdasu *et al.*(4) proposed a logistic regression model for detecting malicious users from URL data. Suhuan and Xiaojun(5) proposed a multidimensional feature fusion method based on logistic regression and XGBoost for detecting malicious applications. Gao *et al.*(6) proposed a semisupervised transfer learning approach for detecting malware in cloud. Wadkar *et al.*(7) used feature rankings based on the weights of a linear support vector machine (SVM) to recognize evolutionary changes within malware families. Singh *et al.*(8) proposed combining scores from morphing strategies and used an SVM to obtain significant classification results. Qi *et al.*(9) used the adversarial learning framework for unsupervised domain adaptation to enable gradient-boosting decision trees to learn domain-invariant features and thus prevent performance degradation in the target domain. Li *et al.*(10) used word frequency and two deep learning algorithms to extract opcode features and probability features from .ASM and byte files; a convolutional neural network (CNN) was used to classify the fused samples. Chen *et al.*(11) proposed four easy-to-extract and small-scale features for classifying malware families and used automatic machine learning to search for the optimal model and hyperparameters for each feature and their combinations. Düzgün *et al.*(12) used histogram-based gradient boosting, random forest, SVM, and XGBoost models for analyzing the multiclass malware classification performance of the balanced and imbalanced versions of these models.

Regarding image-sensing-based methods of malware classification, some researchers have proposed the conversion of byte files into grayscale images and the use of deep learning to perform malware classification. Garcia and Muga II(13) converted a malware binary into an image and used the random forest algorithm to classify various malware families. Gao *et al.*(14) proposed an effective malware classification framework based on malware visualization and semisupervised learning. They used the local binary pattern method for feature extraction and employed a feature fusion method to fuse local and global features, which reduced the time required for feature extraction and improved the relevance of the extracted features. They also proposed an improved collaborative learning algorithm for continually training and optimizing the classifier. Nataraj *et al.*(15) proposed a simple and effective method for visualizing and classifying malware using image-sensing techniques. To determine malware image features, a wavelet decomposition method was applied to an image. Kalash *et al.*[16] proposed a CNN-based architecture for classifying malware samples. They converted malware binaries into grayscale images and subsequently trained a CNN for classification. They also selected the architecture of the CNN through trial and error.

To solve the malware family classification problem by using image-sensing technology, in this paper we propose a Taguchi-based deep learning network (TDLN) with optimization of the parameter combination. The deep learning network (DLN) comprises three convolutional layers, three rectified linear unit (ReLU) activation functions, three max pooling layers, and a fully connected layer. The Taguchi method is used to determine the optimal combination of influential factors, and the orthogonal table design is used to conduct experiments. The major contributions of this study are as follows.

1. The TDLN is proposed for solving malware family classification problems.
2. The optimal combination of parameters—such as the convolution kernel size, number of paddings, and number of filters—for a DLN is obtained using the Taguchi method.
3. The Taguchi method is used to reduce the number of experiments and improve the accuracy of malware family classification.

The remainder of this paper is organized as follows. In Sect. 2, we describe the framework of the proposed TDLN classification system and Taguchi method. Section 3 presents the experimental results obtained when using the proposed TDLN. Finally, the conclusions of this study and recommendations for future research are presented in Sect. 4.

2. Materials and Methods

A TDLN with parameter combination optimization is proposed to improve the performance of malware classification. The framework of the proposed TDLN classification system is illustrated in Fig. 1. In Sect. 2.1, we describe the proposed classification system. The Taguchi method used to optimize the combination of DLN parameters is detailed in Sect. 2.2.

2.1 DLN architecture

A DLN architecture (Fig. 2) comprises four parts: convolution layers, activation functions, pooling layers, and a fully connected layer. The DLN architecture used in this study comprises three convolutional layers, three ReLU activation functions, three max pooling layers, and a fully connected layer.

The parameters of the DLN adopted in this study are presented in Table 1. Images with a size of 128 × 128 × 3 pixels are input to the network structure. The first convolutional layer contains 16 filters with a convolution kernel size of 3 × 3 pixels. The second convolutional layer contains 32 filters with a convolution kernel size of 3 × 3 pixels. The third convolutional layer contains 64 filters with a convolution kernel size of 3 × 3 pixels. Full zero padding and one stride are adopted in these convolutional layers. Three max pooling layers are inserted between the three convolutional layers. To categorize samples as being of one of 25 types, the fully connected layer comprises 256 input nodes, 80 hidden nodes, and 25 output nodes.

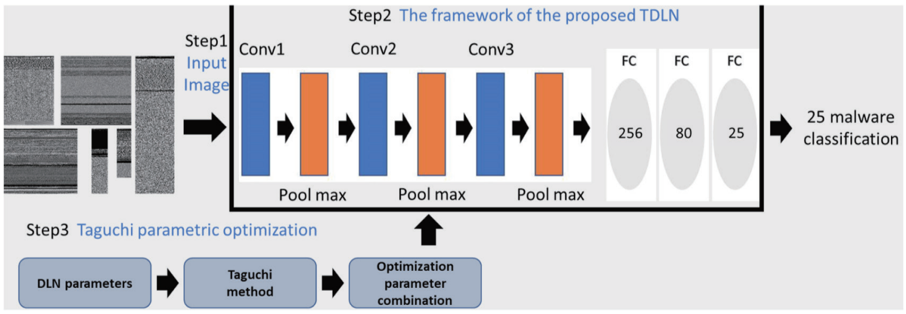

Step1 Input Image
Step2 The framework of the proposed TDLN
Conv1
Conv2
Conv3
FC
FC
FC
256
80
25
Pool max
Pool max
Pool max
25 malware classification
Step3 Taguchi parametric optimization
DLN parameters
Taguchi method
Optimization parameter combination

Fig. 1. (Color online) Framework of the proposed TDLN classification system.

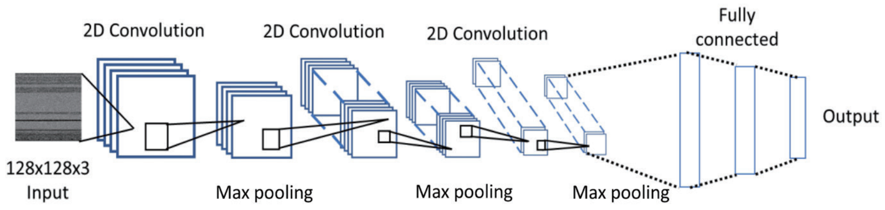


Fig. 2. (Color online) Basic structure of DLN.

Table 1
Parameters of the DLN adopted in this study.

Layer	Image Size	Kernel Size	Stride	Padding	Filter
Input	128 × 128 × 3				
Convolution layer 1		3 × 3	1	0	16
Max pooling layer 1		2 × 2	2 × 2		
Convolution layer 2		3 × 3	1	0	32
Max pooling layer 2		2 × 2	2 × 2		
Convolution layer 3		3 × 3	1	0	64
Max pooling layer 3		2 × 2	2 × 2		
Fully connected layer		1 × 1	1	0	256
Hidden layer					80
Output layer					25

2.2 Taguchi method

The Taguchi method is used to determine the optimal combination of DLN parameters. The goal of the Taguchi method is to achieve the highest possible quality at the lowest cost. An effective combination can be identified through fewer experiments when using the Taguchi method than when using the full factorial method. Although the Taguchi method is not as accurate as the full factorial method in calculating the optimal combination, it can identify the optimization trend through only a few experiments and is thus far more feasible than the full factorial method. A flowchart of the Taguchi method is presented in Fig. 3.

The signal-to-noise (*S*/*N*) ratio is the ratio between the desired information in a signal and the undesired background noise signal. In the Taguchi method, which is an experimental design

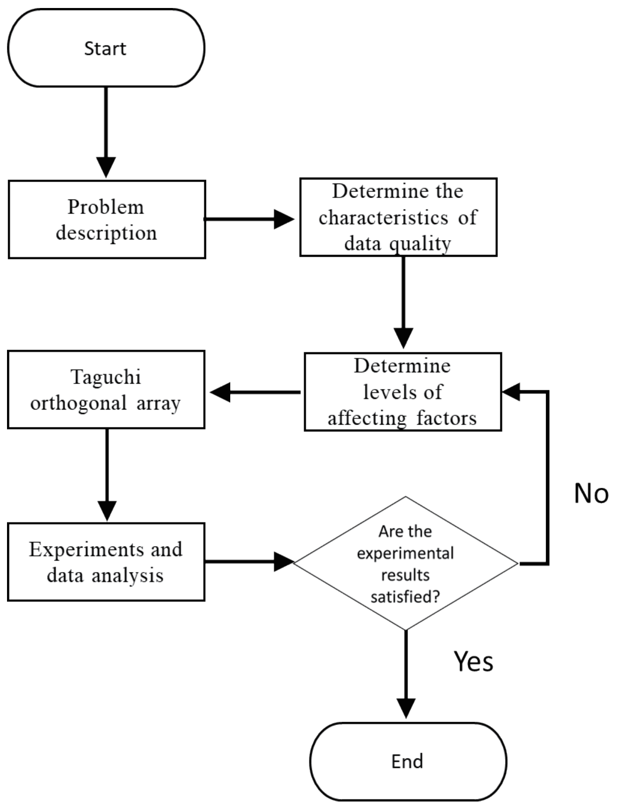

Start
Problem description
Determine the characteristics of data quality
Determine levels of affecting factors
Taguchi orthogonal array
Experiments and data analysis
Are the experimental results satisfied?
No
Yes
End

Fig. 3. Flowchart of Taguchi method.

method, a Taguchi orthogonal array (OA) is designed to collect experimental data for analysis. In this method, a selected subset of combinations of multiple influential factors at multiple levels is used. In general, the *S*/*N* ratio is used as the optimization criterion and analyzed in terms of three performance characteristics: larger-is-better, nominal-is-better, and smaller-is-better.(17) The larger-is-better performance characteristic was considered in this study,

$$S/N = -10\log\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{y_i^2}\right), \tag{1}$$

where *n* represents the total number of replications per test run and *y* represents the classification output of the system.

3. Experimental Results

3.1 Data sets and experimental settings

In this study, we employed the Malimg data set,(15) which comprises 9339 malware grayscale images. Each malware image in the data set belongs to one of 25 malware classes. In the conducted experiment, 70% of malware images in a class were randomly selected for training, and the remaining 30% were used for testing. The initial parameter settings of the TDLN are presented in Table 2.

Table 2
Initial parameter settings of the TDLN.

Batch size	64
Epochs	20
Learning rate	0.00001

3.2 Taguchi experimental design

Nine influential factors with various levels were considered in this study: Conv1_Kernel size, Conv1_Filter, Conv1_Padding, Conv2_Kernel size, Conv2_Filter, Conv2_Padding, Conv3_ Kernel size, Conv3_Filter, and Conv3_Padding. Table 3 details the parameters related to the influential factors of the TDLN, where A–C, D–F, and G–I are the parameters of the first, second, and third convolutional layers, respectively.

The determined factor levels were then used for Taguchi experimental design by using Minitab statistical software (Minitab Inc., Pennsylvania, USA). In this study, the L36 OA of the Taguchi method was used to optimize the combination of TDLN parameters. If the full factorial method were used for experiments, the total number of experiments would be $3^9 = 19683$. By contrast, with the Taguchi method based on an OA table, only 36 experiments need be performed. Although the full factorial method can find the most efficient parameter combination, this method is less feasible and more time-consuming than the Taguchi method. Table 4 presents the experimental results obtained using the L36 OA of the Taguchi method. The first row of Table 4 presents the experimental parameters, the accuracy of the TDLN, and the *S*/*N* ratio for each experiment. For the experimental parameters listed in Table 4, C1_K indicates the choice of matrix size. For example, if C1_K is 3, it indicates a kernel size of 3 × 3 in the first convolutional layer. C1_F indicates the number of convolutional filters in the first convolutional layer.

To determine the optimal experimental results, the *S*/*N* ratio for each factor and level was calculated using an ideal mass function (i.e., higher is better) and the regression model's accuracy. If the accuracy was high in the optimization process, the *S*/*N* ratio was high. According to the results obtained from 36 experiments (Table 4), the maximum accuracy (98.59%) was achieved in experiment 27, whereas the minimum accuracy (96.10%) was achieved in experiment 30. Analysis of variance (ANOVA) was used to determine the significant parameters of the TDLN.(18) If the level of an influential factor changes, the degree of optimization of the parameter combination might be affected. If the obtained influential factors improved the accuracy of the TDLN, the experiment was terminated; otherwise, the levels of the influential factors were redetermined.

The obtained *S*/*N* ratios are presented in Table 5. Table 5 and Fig. 4 indicate that the *S*/*N* ratio based on the yield extraction method affected each level of the parameters. The significant factors were determined using the difference between the highest and lowest *S*/*N* ratio for each influential factor. The experimental results indicated that factor C3_P had a large delta value of 6.77%, which indicated that this factor had the most significant effect on the regression model's accuracy. The order of parameters in terms of importance was as follows: C3_P > C2_P > C3_F > C3_K > C1_K > C1_F > C1_P > C2_F > C2_K. Thus, the best levels were A (level 1 for

Table 3
Parameters related to the influential factors of the TDLN.

No.	Influential factors	Level 1	Level 2	Level 3
A	Conv1_Kernel size (C1_K)	3 × 3	5 × 5	
B	Conv1_Filter (C1_F)	6	16	32
C	Conv1_Padding (C1_P)	0	1	
D	Conv2_Kernel size (C2_K)	3 × 3	5 × 5	
E	Conv2_ Filter (C2_F)	16	32	64
F	Conv2_Padding (C2_P)	0	1	
G	Conv3_Kernel size (C3_K)	3 × 3	5 × 5	
H	Conv3_ Filter (C3_F)	32	64	128
I	Conv3_Padding (C3_P)	0	1	

Table 4
Experimental results obtained using the L36 OA of the Taguchi method.

No.	C1_K	C1_F	C1_P	C2_K	C2_F	C2_P	C3_K	C3_F	C3_P	Accuracy (%)	*S*/*N* ratio
1	3	6	0	3	16	0	3	32	0	97.19	−0.247270
2	3	16	0	3	32	0	3	64	0	97.17	−0.249058
3	3	32	0	3	64	0	3	128	0	97.37	−0.231199
4	3	6	0	3	16	0	3	32	1	96.96	−0.268148
5	3	16	0	3	32	0	3	64	1	97.68	−0.204183
6	3	32	0	3	64	0	3	128	1	97.48	−0.221690
7	3	6	1	3	16	1	5	64	0	97.64	−0.207445
8	3	16	1	3	32	1	5	128	0	98.18	−0.159834
9	3	32	1	3	64	1	5	32	0	97.61	−0.210410
10	3	6	1	5	16	1	3	128	0	98.21	−0.156591
11	3	16	1	5	32	1	3	32	0	97.89	−0.184938
12	3	32	1	5	64	1	3	64	0	98.35	−0.144218
13	3	6	0	5	32	1	5	128	1	97.86	−0.188192
14	3	16	0	5	64	1	5	32	1	98.00	−0.175774
15	3	32	0	5	16	1	5	64	1	97.52	−0.218423
16	3	6	1	5	32	0	5	128	1	97.73	−0.199442
17	3	16	1	5	64	0	5	32	1	96.91	−0.272628
18	3	32	1	5	16	0	5	64	1	97.87	−0.186712
19	5	6	1	3	32	0	5	32	0	96.18	−0.338606
20	5	16	1	3	64	0	5	64	0	93.89	−0.547922
21	5	32	1	3	16	0	5	128	0	97.41	−0.228226
22	5	6	0	3	32	1	5	64	1	97.57	−0.213674
23	5	16	0	3	64	1	5	128	1	98.14	−0.163079
24	5	32	0	3	16	1	5	32	1	97.94	−0.180502
25	5	6	1	3	64	1	3	64	1	98.43	−0.137744
26	5	16	1	3	16	1	3	128	1	98.53	−0.128336
27	5	32	1	3	32	1	3	32	1	98.59	−0.123343
28	5	6	0	5	64	0	5	64	0	96.73	−0.288776
29	5	16	0	5	16	0	5	128	0	97.12	−0.253528
30	5	32	0	5	32	0	5	32	0	96.10	−0.345532
31	5	6	1	5	64	0	3	128	1	97.94	−0.180502
32	5	16	1	5	16	0	3	32	1	97.62	−0.208927
33	5	32	1	5	32	0	3	64	1	98.28	−0.150402
34	5	6	0	5	64	1	3	32	0	97.25	−0.242506
35	5	16	0	5	16	1	3	64	0	96.14	−0.341918
36	5	32	0	5	32	1	3	128	0	97.03	−0.261581

Table 5
S/*N* ratios for each level and the optimal parameters.

Level	Influential factors								
	A	B	C	D	E	F	G	H	I
	C1_K	C1_F	C1_P	C2_K	C2_F	C2_P	C3_K	C3_F	C3_P
1	−0.2070	−0.2224	−0.2386	−0.2256	−0.2188	−0.2386	−0.2046	−0.2332	−0.2578
2	−0.2408	−0.2408	−0.2092	−0.2223	−0.2182	−0.1910	−0.2433	−0.2409	−0.1901
3		−0.2085			−0.2347			−0.1977	
Delta	0.0338	0.0323	0.0294	0.0033	0.0165	0.0658	0.0387	0.0432	0.0677
Rank	5	6	7	9	8	2	4	3	1
Best level	1	3	2	2	2	2	1	3	2
Optimal parameter	3	32	1	5	32	1	3	128	1

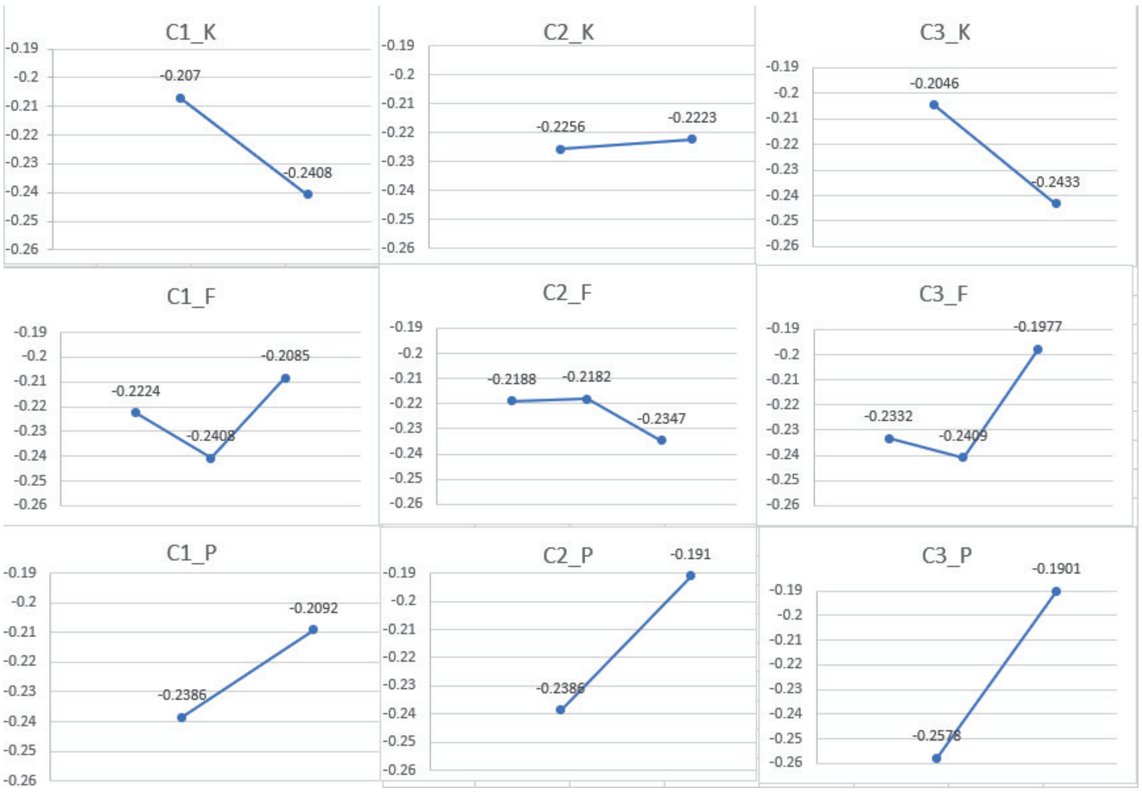


Fig. 4. (Color online) Response graph for the *S*/*N* ratios of factor levels.

C1_K), B (level 3 for C1_F), C (level 2 for C1_P), D (level 2 for C2_K), E (level 2 for C2_F), F (level 2 for C2_P), G (level 1 for C3_K), H (level 3 for C3_F), and I (level 2 for C3_P). Therefore, the optimal parameter combination was C1_K = 3, C1_F = 32, C1_P = 1, C2_K = 5, C2_F = 32, C2_P = 1, C3_K = 3, C3_F = 128, and C3_P = 1.

The ANOVA results presented in Table 6 indicate the degree of influence of each factor and the optimal parameter combination. In this table, the degrees of freedom (DOFs), sum of squares (SS), F ratio of factor A (FA), and percentage contribution (PC; %) are detailed. The PCs of factors F and I were 29.374% and 30.991%, respectively; thus, C2_P and C3_P strongly influenced the optimization results.

Table 6
Results of ANOVA.

Factor	DOF	SS	FA	PC (%)
A	1	0.0103	2.66	7.749
B	2	0.0063	0.81	4.747
C	1	0.0078	2.00	5.843
D	1	0.0001	0.03	0.007
E	2	0.0020	0.27	1.575
F	1	0.0389	10.05	29.374
G	1	0.0134	3.47	10.126
H	2	0.0127	1.64	9.588
I	1	0.0411	10.63	30.991
Sum	35	0.221217		100

A confusion matrix is employed for analyzing image classification results. It can be used to calculate accuracy, sensitivity, and specificity. A confusion matrix comprises four elements: true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*), as presented in Table 7.

By using a confusion matrix, the classification performance of the proposed TDLN after Taguchi parameter optimization was determined in terms of accuracy, recall, and precision. The accuracy, recall, and precision are expressed as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2}$$

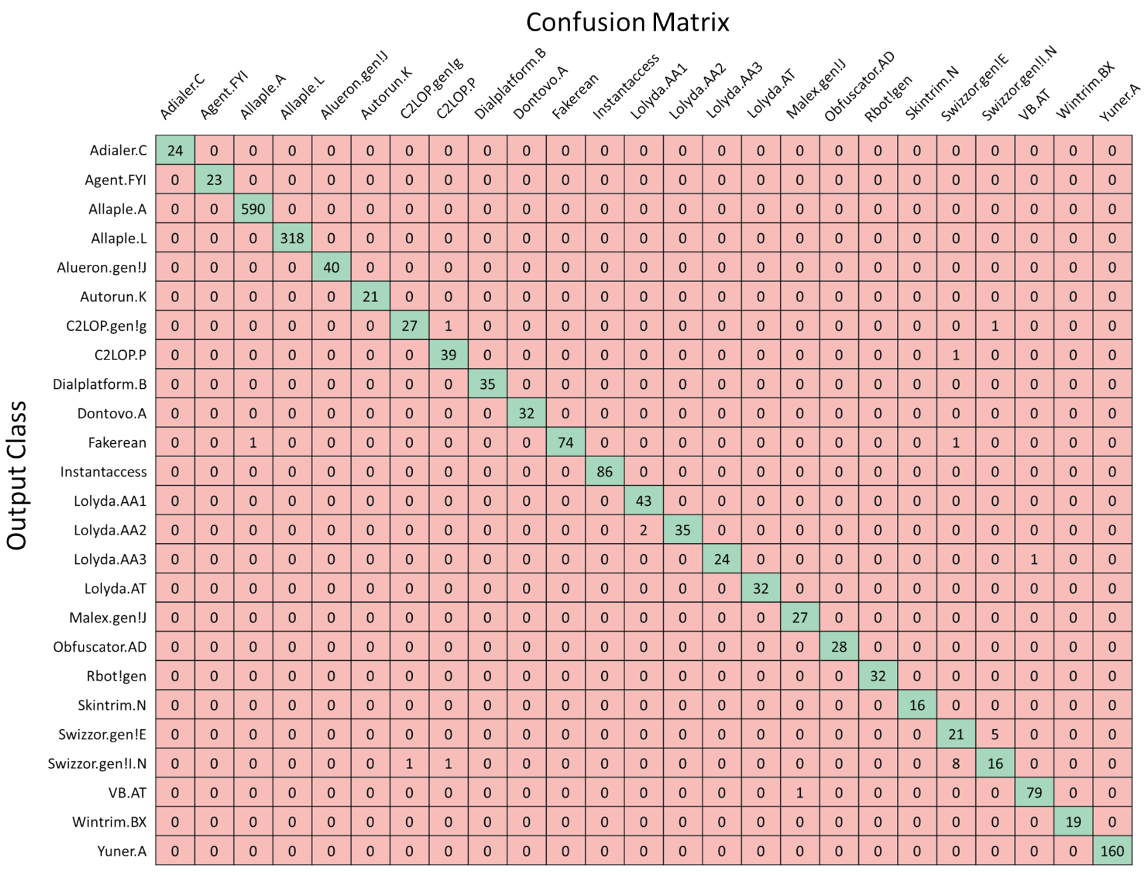
$$Recall = \frac{TP}{TP + FN}, \tag{3}$$

$$Precision = \frac{TP}{TP + FP}. \tag{4}$$

Accuracy represents the rate of correct malware family classification across the entire data set; sensitivity represents the network's ability to classify malware images into the correct family; and specificity indicates the extent to which the network correctly identifies that a malware image does not belong to a particular family. Figure 5 presents the confusion matrix of the malware family classification with the proposed TDLN after the Taguchi optimization of the parameter combination. Table 8 indicates that the accuracy, precision, and recall of the proposed TDLN were 98.71, 96.90, and 96.78%, respectively. Moreover, the accuracy, precision, and recall of the original DLN were 96.68, 91.31, and 90.69%, respectively. Thus, the precision, recall, and accuracy of the TDLN with parameter optimization were 2.03, 5.59, and 6.09% higher, respectively, than those of the original DLN.

Table 7
Confusion matrix.

	Positive	Negative
True	*TP*	*TN*
False	*FP*	*FN*


Confusion Matrix
Output Class

	Adialer.C	Agent.FYI	Allaple.A	Allaple.L	Alueron.gen!J	Autorun.K	C2LOP.gen!g	C2LOP.P	Dialplatform.B	Dontovo.A	Fakerean	Instantaccess	Lolyda.AA1	Lolyda.AA2	Lolyda.AA3	Lolyda.AT	Malex.gen!J	Obfuscator.AD	Rbot!gen	Skintrim.N	Swizzor.gen!E	Swizzor.gen!I.N	VB.AT	Wintrim.BX	Yuner.A
Adialer.C	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Agent.FYI	0	23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Allaple.A	0	0	590	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Allaple.L	0	0	0	318	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Alueron.gen!J	0	0	0	0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Autorun.K	0	0	0	0	0	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C2LOP.gen!g	0	0	0	0	0	0	27	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
C2LOP.P	0	0	0	0	0	0	0	39	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Dialplatform.B	0	0	0	0	0	0	0	0	35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Dontovo.A	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Fakerean	0	0	1	0	0	0	0	0	0	0	74	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Instantaccess	0	0	0	0	0	0	0	0	0	0	0	86	0	0	0	0	0	0	0	0	0	0	0	0	0
Lolyda.AA1	0	0	0	0	0	0	0	0	0	0	0	0	43	0	0	0	0	0	0	0	0	0	0	0	0
Lolyda.AA2	0	0	0	0	0	0	0	0	0	0	0	0	2	35	0	0	0	0	0	0	0	0	0	0	0
Lolyda.AA3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	0	0	0	0	0	0	0	1	0	0
Lolyda.AT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0
Malex.gen!J	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0
Obfuscator.AD	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	28	0	0	0	0	0	0	0
Rbot!gen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0
Skintrim.N	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0
Swizzor.gen!E	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21	5	0	0	0
Swizzor.gen!I.N	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	8	16	0	0	0
VB.AT	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	79	0	0
Wintrim.BX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0
Yuner.A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160

Fig. 5. (Color online) Confusion matrix of malware family classification by the proposed TDLN.

Table 8
Performances of the original DLN and TDLN.

Method	Accuracy (%)	Precision (%)	Recall (%)
DLN	96.68	91.31	90.69
TDLN	98.71	96.90	96.78

3.3 Comparison of various methods

The malware family classification performance of the proposed TDLN was evaluated by comparing the classification performance of this network and some other methods for the Malimg data set.[9,13–16] The accuracies of the various methods are presented in Table 9. The experimental results indicate that the proposed TDLN has higher accuracy than the other methods.

Table 9
Accuracies of various methods.

Method		Accuracy (%)
Logistic regression		92.69
KNN		93.31
SVM		93.38
Qi *et al.*(9)		93.37
Garcia and Muga II(13)		95.62
Gao *et al.*(14)		97.95
Nataraj *et al.*(15)		97.18
Kalash *et al.*(16)		98.52
Proposed method	DLN	96.68
	TDLN	98.71

4. Conclusions

In this study, a TDLN with parameter combination optimization was proposed for malware family classification. The architecture of the DLN comprised three convolutional layers, three max pooling layers, and a fully connected layer. The Taguchi method was used to determine the optimal combination of influential factors for the TDLN and to improve the accuracy of malware family classification. An OA was used to conduct experiments. The experimental results indicated that the accuracies of the original DLN and TDLN were 96.68% and 98.71%, respectively, and that on average, the accuracy of the TDLN was 2.03% higher than that of the original DLN.

Some parameters of the proposed TDLN, such as the batch size and learning rate, are generally difficult to determine and affect experimental results. Therefore, these parameters had to be predetermined in this study. To prevent the preset parameter problem, the random search or Gaussian optimization method(19) will be used to determine the optimal values of these parameters in a future study.

References

1 T. Alsmadi and N. Alqudah: 2021 Int. Conf. Information Technology (2021) 371−376. https://doi.org/10.1109/ICIT52682.2021.9491765
2 K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov: Int. Conf. Detection of Intrusions and Malware, and Vulnerability Assessment (2008) 108–125. https://doi.org/10.1007/978-3-540-70542-0_6
3 T. A. Assegie: Int. J. Comput. Eng. Res. Trends **8** (2021) 46. https://doi.org/10.22362/ijcert/2021/v8/i02/v8i022
4 R. Chiramdasu, G. Srivastava, S. Bhattacharya, P. K. Reddy, and T. R. Gadekallu: 2021 IEEE Int. Conf. Omni-Layer Intelligent Systems (IEEE, 2021) 1−6. https://doi.org/10.1109/COINS51742.2021.9524269
5 L. Suhuan and H. Xiaojun: 2019 IEEE 10th Int. Conf. Software Engineering and Service Science (2019) 528−532. https://doi.org/10.1109/ICSESS47205.2019.9040851
6 X. Gao, C. Hu, C. Shan, Z. Niu, and H. Xie: J. Inf. Secur. Appl. **55** (2020) 102661. https://doi.org/10.1016/j.jisa.2020.102661
7 M. Wadkar, F. D. Troia, and M. Stamp: Expert Syst. Appl. **143** (2020) 113022. https://doi.org/10.1016/j.eswa.2019.113022.
8 T. Singh, F. D. Troia, C. A. Visaggio, T. H. Austin, and M. Stamp: J. Comput. Virol. Hacking Tech. **12** (2016) 203. https://doi.org/10.1007/s11416-015-0252-0
9 P. Qi, W. Wang, L. Zhu, and S. K. Ng: Proc. 30th ACM Int. Conf. Information & Knowledge Management (2021) 1457–1466. https://doi.org/10.1145/3459637.3482400

10 L. Li, Y. Ding, B. Li, M. Qiao, and B. Ye: Alexandria Eng. J. **61** (2022) 91. https://doi.org/10.1016/j.aej.2021.04.076
11 Z. Chen, E. Brophy, and T. Ward: arXiv:2201.07649 (2021). https://doi.org/10.48550/arXiv.2201.07649
12 B. Düzgün, A. Çayır, F. Demirkıran, C. N. Kayha, B. Gençaydın, and H. Dağ: arXiv:2111.15205 (2021). https://doi.org/10.48550/arXiv.2111.15205
13 F. C. C. Garcia and F. P. Muga II: arXiv:1609.07770 (2016). https://doi.org/10.48550/arXiv.1609.07770
14 T. Gao, L. Zhao, X. Li, and W. Chen: Math. Biosci. Eng **18** (2021) 5995. https://doi.org/10.3934/mbe.2021300
15 L. Nataraj, S. Karthikeyan, G. Jacob, and B. Manjunath: Proc. 8th Int. Symp. Visualization for Cyber Security (2011) 1–7. https://doi.org/10.1145/2016904.2016908
16 M. Kalash, M. Rochan, N. Mohammed, N. D. B. Bruce, Y. Wang, and F. Iqbal: 2018 9th IFIP Int. Conf. New Technologies, Mobility and Security (2018) 1–5. https://doi.org/10.1109/NTMS.2018.8328749
17 A. Özakın and F. Kaya: J. Sol. Energy **197** (2020) 199. https://doi.org/10.1016/j.solener.2019.12.077
18 F. N. Idris, M. M. Nadzir, and S. R. A. Shukor: J. Environ. Chem. Eng. **8** (2020) 103766. https://doi.org/10.1016/j.jece.2020.103766
19 Q. Meng, S. Wang, and S. H. Ng: arXiv:2107.03217 (2021). https://doi.org/10.48550/arXiv.2107.03217

About the Authors

Cheng-Jian Lin received his B.S. degree in electrical engineering from Ta Tung Institute of Technology, Taipei, Taiwan, R.O.C., in 1986 and his M.S. and Ph.D. degrees in electrical and control engineering from National Chiao Tung University, Taiwan, R.O.C., in 1991 and 1996, respectively. Currently, he is a chair professor of the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung, Taiwan, R.O.C., and dean of Intelligence College, National Taichung University of Science and Technology, Taichung, Taiwan, R.O.C. His current research interests are machine learning, pattern recognition, intelligent control, image processing, intelligent manufacturing, and evolutionary robots.
(cjlin@ncut.edu.tw)

Xin-You Lin received his B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung, Taiwan, in 2016 and 2018, respectively. He is currently pursuing his Ph.D. degree with Prospective Technology of Electrical Engineering and Computer Science, National Chin-Yi University of Technology, Taichung, Taiwan. His current research interests include deep learning, neural fuzzy systems, intelligent manufacturing, and computer vision and applications. (kamisama0406@gmail.com)

Jyun-Yu Jhang received his B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung, Taiwan, in 2013 and 2015 and his Ph.D. degree from Institute of Electrical and Control Engineering in 2021. Currently, he is an assistant professor in the Computer Science & Information Engineering Department, National Taichung University of Science and Technology, Taichung, Taiwan. His current research interests include fuzzy logic theory, type-2 neural fuzzy systems, evolutionary computation, machine learning, computer vision, and applications. (jyjhang@nutc.edu.tw)