Sensors and Materials, Vol. 30, No. 3 (2018) 385–396
MYU Tokyo

S & M 1500

On Neural Networks for Biometric Authentication Based on Keystroke Dynamics

Chu-Hsing Lin,* Jung-Chun Liu, and Ken-Yu Lee

Department of Computer Science, Tunghai University,
No. 1727, Sec. 4, Taiwan Boulevard, Xitun District, Taichung 40704, Taiwan

(Received July 24, 2017; accepted November 2, 2017)

Keywords: biometric authentication, keystroke dynamics, machine learning, convolutional neural network, GPU parallel computing

Nowadays, passwords have become closely associated with our daily activities. However, the development of technology also increases the risk of password leak. For example, the graphics processing unit (GPU)-parallel-computing-based brute force attack and birthday attack algorithms have greatly reduced password security; in addition, passwords are usually transmitted through wired or wireless communication media and thus are vulnerable to attack and easily exposed to illegal users. In this study, we propose a biometric authentication method to identify and block illegal users, even if the entire password is exposed. Our method simultaneously records scan codes and the keystroke sequence of passwords; furthermore, by deep learning of convolutional neural networks (CNNs), it can effectively distinguish legal users from illegal users. We first compare recognition rates between the CNN and the neural network (NN) and prove that the CNN is the better choice. The experimental results show that the proposed CNN model can block all illegal users even if the password is known by them. By using equal amounts of password data from legal and illegal users, the average login failure rate of legal users is 6%, and they can always enter passwords again to be admitted. Finally, by GPU parallel computing, we further accelerate the system performance by 4.45 times.

1. Introduction

With the rapid development of technology, the use of passwords has become inseparable in our daily life; however, the development of technology also accelerates the cracking of passwords. For example, brute force methods and birthday attack algorithms using the graphics processing unit (GPU) parallel computing have considerably decreased the security of passwords. Moreover, information of any complex passwords could be leaked owing to carelessness or negligence of users. Additionally, smartwatches equipped with motion recognition sensors worn by users may pose security threats of password leakage. By analyzing the patterns of the wrist movements exposed to motion recognition sensors, attackers can estimate the passwords entered by the users.(1)

*Corresponding author: e-mail: chlin@thu.edu.tw
http://dx.doi.org/10.18494/SAM.2018.1757

ISSN 0914-4935
http://myukk.org/

In this study, we present an identification method that can block most illegal users who know the complete password. We use a person's typing habits, i.e., keystroke dynamics, as biometric features, which are used as input for a machine learning system that can differentiate between legitimate users and illegal users.[2] At first glance, the differences in typing habits between individuals are extremely small, but, in fact, this is not the case. Users can have different typing habits in typing passwords; these habits could be differences between two key-press times or the use of left shift or right shift keys, and they increase distinctiveness for typing passwords.[3,4] By taking advantage of those unique habits, one can distinguish between legal and illegal users. However, these biological characteristics are unlikely to be seen by the naked eye or with general decision-making techniques in programming. Hence, in this study, we use the powerful capability of machine learning, and via several numbers of learning processes, the proposed system can ultimately distinguish between legal and illegal users.[5–10]

Machine learning is a kind of artificial intelligence; it finds an approximate solution of a problem through reasoning and induction, and modifies the solution from learning experiences, such that the solution will gradually approach the correct value. In this study, we discuss two different models of machine learning: neural networks (NNs) and convolutional neural networks (CNNs). Both models can use input features to analyze or predict the results. The difference is that when performing feature analysis, the CNN cascades multiple features to let features associate with one another, and thus it can significantly enhance the accuracy of the analysis or prediction.[11–13] However, the CNN often leads to overfitting because its learning ability is very strong. Overfitting refers to the distortion of analysis curves caused by too much learning in the machine learning process, so that the result is not as expected. To avoid this situation, in this paper, we added a dropout regularization condition, which can appropriately ignore parts of the neuron weights and greatly reduce overfitting chances.[14] In the experiments, we compared the execution results in different situations, and identified the best learning model, significantly reducing the risk of accounts being compromised. In addition, we also performed experiments designed for a real world scenario, in which the data of illegal users is limited; the system still has more than 80% accuracy for this setting. Finally, we added GPU parallel computing into our system and significantly optimized the system performance.

2. Background

In this section, we provide overviews of background technologies and related studies.

2.1 NN

An NN is formed by one or more types of neurons, and can be divided into the input layer, hidden layer, and output layer. The input layer does not have neurons. After training and teaching these neurons using different algorithms, the NN, like a human brain, can output expected results.

Figure 1 shows a schematic of an NN, in which x represents a component of the input vector, which has a total of n inputs; w is the link value, i.e., the weight of each link, since the most important task of NNs is learning to adjust the link values through constant training; b is the

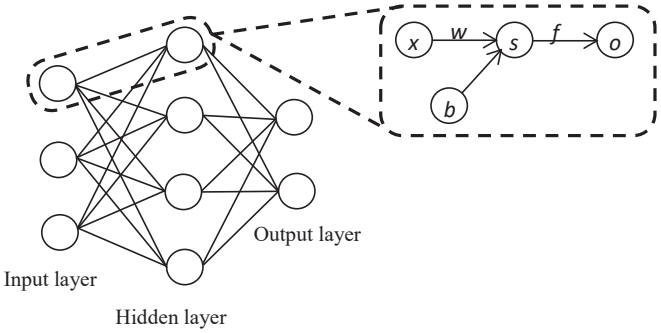

Fig. 1. Schematic of NN.

bias; s is the summation unit, which adds up each value of neuron input timed with the link value; f is activation function, which is a linear or nonlinear function to convert s to the wanted answers; o is the output vector from the output neurons.

In this study, we use two activation functions as follows:

$$f(x) = max(0, w^T x + b)\,. \tag{1}$$

Here, w is the weight of each link, x is a component of the input vector, and b is the bias.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \text{ for } j = 1, ..., K \tag{2}$$

Here, z is a K-dimensional vector input, and j indexes the K-dimensional vector output $\sigma(z)$.

The activation function in Eq. (1) is called the Rectified Linear Unit (ReLU), which can change the threshold of neurons; when the result of any neuron is negative, it converts the result to 0; otherwise, it keeps the raw result. Since neurons with results of 0 will have no effect on the output of the entire NN, this equation can control the activity of each neuron in the NN model.[15]

The softmax activation function, or normalized exponential function, in Eq. (2) uses the natural logarithm conversion to convert the value of the neuron between 0 and 1. As a result, it can compare the value of each neuron, and is important for classification or decision analysis, so in this paper, we use the softmax activation function to distinguish between valid and invalid users.[15]

2.2 CNN

The CNN is a neural-network-based model with excellent performance for image processing; it has largely the same architecture as the NN, except that it has one or more convolutional layers and pooling layers, as shown in Fig. 2.

Convolutional layer: Each convolutional layer consists of several convolution units, which can extract parts of the input arrays and send the extracted arrays to the NN to perform analysis; in contrast to traditional NNs, the convolutional layer enables the NN to read more than one feature at a time.

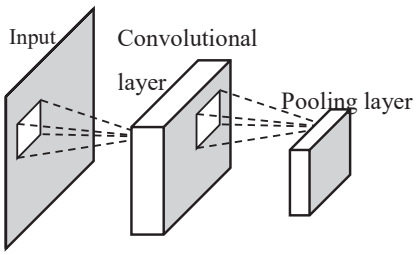

Fig. 2. Schematic of convolutional layer and pooling layer.

Pooling layer: One important concept for CNNs is to compress the input features. The compressing process will keep the main features to narrow the range of features, reduce computational complexity, and highlight the main features.

Figure 2 depicts the relationship between the convolutional layer and the pooling layer. The input image is processed by the convolutional layer and pooling layer to reduce its length and width but increase its thickness. The pooling layer helps retain the features while the length and width of the image are reduced. By adding the convolutional layer and the pooling layer, the whole network no longer uses a feature for a neuron, but several features for a neuron in data analysis; and by changes in weights, the correlation between features can be found, thereby significantly enhancing the accuracy of analysis or prediction. CNNs are widely used in areas such as image processing and natural language analysis.[11,13–17]

2.3 Compute unified device architecture (CUDA)

CUDA is a parallel computing architecture from NVIDIA. It uses the considerable processing power of a GPU to significantly increase operational efficiency. A GPU is composed of several streaming multiprocessors (SMs), and each SM contains many stream processors (SPs). Each SP can handle operations independently to achieve parallel processing of data. CUDA is good at dealing with matrix operations, especially for image processing and video processing; it significantly accelerates computing speed, and is clearly helpful in enhancing performance. The NNs normally have a large number of neurons, and each neuron is an independent operation; hence, one can treat neurons as images and use parallel matrix operations to improve performance.[18,19]

3. Experimental Design

The proposed biometric authentication system consists of two parts: biological features extraction and biological features analysis, as described in detail below.

3.1 Biological features extraction

At this stage, we record the typing actions of the users when they enter passwords on the keyboard, containing not only the scan codes of the keys, but also the time points of key

pressing and releasing, which are used to analyze the sequence and interval of pressed keys. The scan codes of pressed keys are recorded, since each key on the keyboard has a unique scan code, whereas, the same characters, such as digital numbers and function keys (Shift, Ctrl, etc.) may be found at two places on the keyboard.

3.2 Biological features analysis

The proposed system uses the CNN model for machine learning, since the powerful feature association capability of the CNN makes it very suitable for analysis applications with a subtle difference.

We used a total of 64 features, divided into six segments, as shown in detail in Table 1. The duration of each key press time for the 10 keys can be computed by subtracting the values in segment *c* by the corresponding ones in segment *b*. Time gaps between the previous key and current key are recorded in segment *f*. The zero padding in segment *e* is used as a reserved space for the possible length extension of passwords in the future. Besides, in the machine learning process, the computer will find that these zero padding features have no effect on the results no matter how their corresponding weights are changed; hence, the computer will automatically ignore these zero padding features.

The proposed CNN architecture consists of 7 stages A–H, as shown in Fig. 3.

Stage A: This stage is the data input layer; the input data is the retrieved biometric feature array with a size of 1 × 64; the system will transform the input array into a 16 × 4 matrix in analysis.

Table 1
Content of input array features.

Segment	Features	Length
a	Scan codes of 10 keys	10
b	Key pressed time point of 10 keys	10
c	Key released time point of 10 keys	10
d	Check if each of ten inputs of 10 keys matched with the corresponding valid passwords	10
e	Zero padding	15
f	Time gap between previous key and current key	9

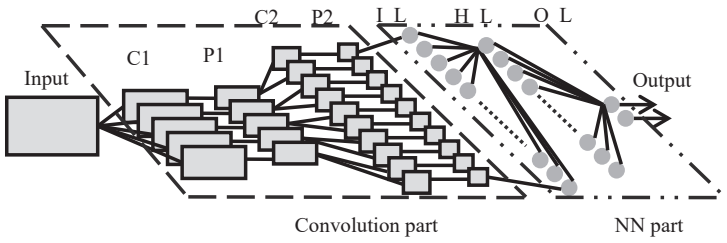

Fig. 3. CNN architecture.

Stage B : This stage consists of the convolutional layer C1, the first convolutional layer used in the model; it extracts features from the input layer; the extracting range is a two-dimensional 2 × 4 matrix; the height is 32. The activation function is ReLU, which replaces negative values with 0 and retains positive values to ensure no negative output.

Stage C: Stage C: This stage consists of the pooling layer P1, the first pooling layer used in the model. The pooling range is a 2 × 2 matrix; it is used to avoid too much data loss from overstriding.

Stage D: This stage consists of the convolutional layer C2, the second convolutional layer used in the model; it is mostly similar to Stage B, except that the range of feature extraction is a 6 × 4 matrix and the transformation height is 64. The height is increased to expand the diversity of features again.

Stage E: This stage consists of the convolutional layer P2, the second pooling layer used in the model; it is similar to Stage C.

Stage F: This stage consists of the input array I–L of the NN in the model. The input values, a two-dimensional array, are outputs of Stage E. The activation function used in this stage is still ReLU. It is worth noting that at this stage a dropout function is added; this function enables machine learning to ignore part (30%) of the neuron weights to avoid the overfitting problem. The output array has a size of 128.

Stage G: This stage consists of the hidden array H–L of the NN in the model. The number of neurons is 128, and the output array size is 2. The softmax activation function is used to set the output value of this stage to be between 0 and 1 as input for classification criteria at the next stage.

Stage H: This is the last stage, i.e., the output layer in the model. At this stage, the machine learning model filters the input values, setting maximum values to be 1, and all the rest to be 0, and thus, it can distinguish valid from invalid users.

As shown in Fig. 4, overfitting is a common problem in machine learning. The main reason for it is overtraining of the machine, resulting in machine learning results being too fit with the training data; this will cause the machine to misjudge new data. Many methods can be used to solve this problem. In this study, we chose to ignore part of weights of the neuron as described in Stage F.

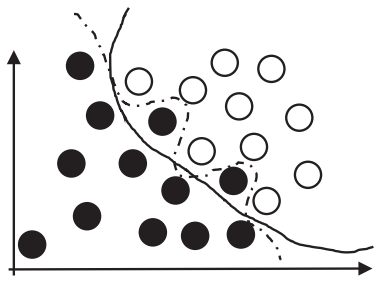

Fig. 4. Schematic of overfitting. The black and white circles represent two types of data; the solid line gives the expected solution and the dotted line shows overfitting.

4. Experimental Results and Analyses

In this section, the experimental results of different application scenarios are presented and discussed. The proposed biometric authentication system can prevent illegal users from successful login even if the complete password has been exposed to them. To verify the reliability of the proposed method, experiments of various testing scenarios were designed. The specifications of the hardware and software used in the test environment are listed in Table 2.

We adopted supervised learning in machine learning. The practice data used in the experiment were recorded from 10 different users, i.e., Users A–J, respectively. The password used by User A had been exposed to the other users, i.e., Users B–J.

All the users repeatedly typed User A's password 100 times. From the 100 recorded password data of each user, we randomly selected 10 recorded password data to use as test data and used the rest as training data. Therefore, there were 100 (10 from User A and 90 from other users) test data and 900 (90 from User A and 810 from other users) training data in total.

We used the false acceptance rate (*FAR*) and the false rejection rate (*FRR*) as the evaluation criteria, in which *FAR* represents the ratio of successful login by an illegal user, while *FRR*, the ratio of unsuccessful login by a legal user.

4.1 Experiment 1

In Experiment 1, the identification rates of the NN and CNN are compared. We used the recorded training data to perform machine learning; the iterations of training were performed 5000 to 25000 times. The experimental results are listed in Table 3. The experimental results show that when the number of trainings is less than 7000 times, both the accuracies of the NN and CNN are 90%; however, the *FRR* is 100%, which means that the authentication system

Table 2
Specifications of the experimental environment.

Device	Specification
CPU	Intel Q8200 2.33 GHz × 4
Memory	3.9 GB
Operating system	Ubuntu 16.04 64-bit
GPU	Nvidia GTX 650Ti
Machine learning tool	Google tensorflow

Table 3
Comparison of identification accuracies (%) of NN and CNN.

Numbers of training	NN accuracy	*FRR*	CNN accuracy	*FRR*
5000	90	100	90	100
7000	90	100	90	100
9000	90	100	92	80
11000	90	100	95	50
13000	90	100	95	50
25000	90	100	99	10

cannot correctly identify the password data typed by the legal user and blocks 100% login trials of the legal user. When the number of trainings increases over 7000 times, the accuracy of the CNN increases; nevertheless, the accuracy of the NN shows no improvement. When the number of trainings increases to 25000 times, the accuracy of the CNN improves to 99%, and its *FRR* is only 10%. From the results, we find that compared with the NN, the CNN has stronger feature identification ability. Hence, we adopted the CNN for biometric authentication in the following experiments.

4.2 Experiment 2

In this experiment, the CNN was used as the analysis tool. The recorded keystroke data of each user were rotationally picked as the training data; in other words, if User A was treated as the valid user, then password data of User A were used as the training data, and so on. The number of trainings was 25000 times. The aim of this experiment is to prove that the biometric feature such as keystroke contains certain characteristics that can be used for identification. The experimental results are listed in Table 4.

From Table 4, we observe that biometric features are effective in identifying the user. The *FRR* of valid User C is worst (40% or 4 out of 10) and it indicates that the sensitivity of the machine learning model tends to dismiss valid users as invalid users. We will further work on adjusting the sensitivity of the machine learning model to decrease the *FRR* of users later in Experiment 4.

4.3 Experiment 3

In this experiment, we used a larger amount of test data in the experiment to verify the reliability of the CNN model to identify password data of the valid user. The test data consisted of the 100 recorded password keystroke data from valid User A and 10 × 9 recorded password keystroke data from the 9 invalid users, i.e., Users B–J. The number of trainings was 25000 and the test had been performed 10 times. The experimental results are listed in Table 5.

Table 4
Comparison of identification accuracies (%) of different valid users.

Valid user	Accuracy	*FAR*	*FRR*
A	99	0	10
B	99	0	10
C	96	0	40
D	99	0	10
E	99	0	10
F	97	0	30
G	98	0	20
H	99	0	10
I	98	0	20
J	98	0	20

Table 5
Identification accuracies (%) of the CNN model with a larger test data set.

Test	*FAR*	*FRR*	Accuracy
1	0	13	93.157
2	0	13	93.157
3	0	13	93.157
4	0	13	93.157
5	0	13	93.157
6	0	13	93.157
7	0	12	93.684
8	0	13	93.157
9	0	13	93.157
10	0	13	93.157

From Table 5, we observe that the proposed CNN model has high identification accuracy. For example, Test 1 has *FAR* = 0% and *FRR* = 13% (13 of 100 test data of valid User A are falsely rejected) and accuracy = (1 − 13/190) × 100% = 93.157%. It shows that the proposed method can effectively block illegal users (100%). However, it is still too sensitive that 13% of legal users are blocked. The legal users can always enter the password to be admitted; the probability of being denied twice is merely 1.69%. We adjusted the training data of the CNN model to enhance its accuracy in the following experiment.

4.4 Experiment 4

In Experiments 1 to 3, the training data consists of 90 password data from User A and 810 password data from other users, i.e., the majority of training data are from illegal users. The high percentage (90%) of illegal users' data may cause the machine to tend to guess users as illegal, resulting in misinterpreting legal users as illegal and thus a higher *FRR*.

In this experiment, the training data consists of 810 password data of User A and 810 password data from other users, i.e., the same amounts of data used in training are from the legal user and the illegal users. The test data is the same as that in Experiment 3, consisting of 100 recorded password keystroke data from valid User A and 10 × 9 recorded password keystroke data from the 9 invalid users, i.e., Users B–J. The number of trainings was 25000 and the test was repeated 10 times. The experimental results are listed in Table 6. We observe that the identification accuracy of the proposed CNN model has been improved to be over 96%, and only 6% of legal users are blocked. We conclude that the structure of the training data affects the learning results.

4.5 Experiment 5

In this experiment, we consider a more realistic scenario, in which the illegal users' data are difficult to acquire and use in the training data. Hence, in this experiment, we use 100 password keystroke data from User A and 90 password keystroke data from one illegal user, e.g., User B. The test data is different from those used in Experiments 3 and 4, since just part of the password keystroke data of Users A and B, and no password keystroke data of Users C–J have been used in the training data. In other words, we tested the accuracy of the CNN model to defend the illegal users without knowing the data of illegal users beforehand. The experimental results are listed in Table 7. We observe that the identification accuracy of the proposed CNN model is reduced to 80% while 27.78% of illegal users are admitted.

Table 6
Identification accuracy (%) of the CNN model with equal amounts of training data from legal and illegal users.

FAR	*FRR*	Accuracy
0	6	96.842

Table 7
Identification accuracy (%) of the CNN model when most illegal users' data are not included in the training process.

FAR	*FRR*	Accuracy
27.78	13	80

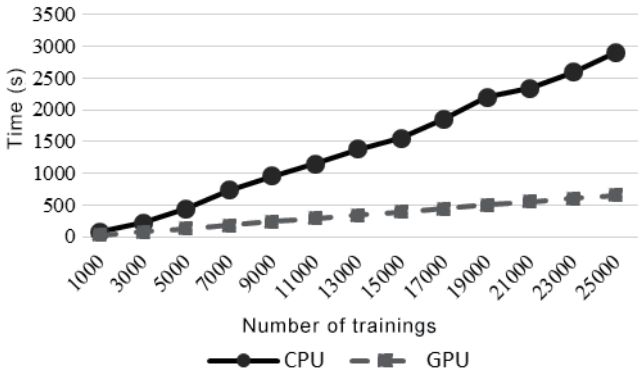

Fig. 5. Operation time of GPU and CPU with various numbers of trainings.

Table 8
Acceleration ratio of GPU over CPU.

Number of trainings	CPU (s)	GPU (s)	Speedup ratio
1000	77.114	27.061	2.849
3000	220.261	79.314	2.777
5000	445.693	131.662	3.385
7000	734.804	183.138	4.012
9000	961.253	235.417	4.083
11000	1149.298	287.582	3.996
13000	1382.495	340.187	4.063
15000	1557.381	391.859	3.974
17000	1851.157	444.683	4.162
19000	2200.956	496.718	4.43
21000	2340.522	549.516	4.259
23000	2598.377	601.671	4.318
25000	2907.79	653.9	4.446

4.6 Experiment 6

We did a performance test in this experiment. For applications with a huge amount of training data and training numbers, the proposed model should be further accelerated. We combined the CNN with GPU parallel computing. The computing times of CPU and GPU and the acceleration ratios of the GPU over the CPU are plotted in Fig. 5 and listed in Table 8. We observe that a maximum of 4.45 acceleration ratio is achieved when the number of trainings is 25000.

5. Conclusions

In this study, we show that the CNNs model for biometric authentication based on keystroke dynamics can greatly enhance the strength of passwords against attacks such as the brute force attack. Even if a user password has been leaked, it still can block 100% of illegal users from login if the password keystroke data of suspected users are used in the training data, so that the account of the legal user is highly protected. Although the *FRR* of legal users is 13%, any legal user can enter the password again when he/she is falsely blocked to be admitted. Moreover, when the structure of the training data is adjusted to let the CNN model be less sensitive in dismissing legal users, the *FRR* of legal users is improved to 6% and the identification accuracy of the CNN model is enhanced to 97%. In addition, we consider a more realistic scenario, in which the password keystroke data of most illegal users are not included in the training. The CNN model can still defend 72% of illegal users, and the identification accuracy of the CNN model is 80%. Finally, considering that future applications will have great demands of computation performance in the face of a huge amount of data and number of trainings, we combine the GPU parallel computing into our CNN model and obtain acceleration ratios about 4.45 times, making the proposed approach highly practical and feasible.

Acknowledgments

This research was partly supported by the Ministry of Science and Technology, Taiwan, under grant number MOST 105-2221-E-029-012-MY2.

References

1 J. Kim and J. M. Youn: Symmetry **9** (2017) 21.
2 M. T. Hagan, H. B. Demuth, and M. Beale: Neural Network Design (PWS Publishing Co., Boston, 1996).
3 F. Monrose and A. D. Rubin: Future Gener. Comput. Syst. **16** (2000) 351.
4 A. Ceffer and J. Levendovszky: Proc. 2016 IEEE 17th Int. Symp. Computational Intelligence and Informatics (CINTI) (2016) 105.
5 Y. Zhang, G. Chang, L. Liu, and J. Jia: 2010 4th Int. Conf. Genetic and Evolutionary Computing (ICGEC) (2010) 578.
6 K. Buza and D. Neubrandt: 2016 IEEE 11th Int. Symp. Applied Computational Intelligence and Informatics (SACI) (2016) 453.
7 P. Chairunnanda, N. Pham, and U. Hengartner: 2011 IEEE 3d Int. Conf. Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE 3rd Int. Conf. Social Computing (SOCIALCOM) (2011) 974.
8 P. Kang and S. Cho: Inf. Sci. **308** (2015) 72.
9 E. Alpaydin: Introduction to Machine Learning (The MIT Press, Cambridge, 2010) 2nd ed.
10 P. S. Teh, A. B. J. Teoh, and S. Yue: Sci. World J. **2013** (2013) 408280.
11 J. Lemley, S. Bazrafkan, and P. Corcoran: IEEE Consum. Electron. Mag. **6** (2017) 48.
12 P. Louridas and C. Ebert: IEEE Software **33** (2016) 110.
13 Convolutional Neural Networks (LeNet)-DeepLearning 0.1 documentation. LISA Lab (2017).
14 N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov: J. Mach. Learn. Res. **15** (2014) 1929.
15 A. Krizhevsky, I. Sutskever, and G. Hinton: Neural Information Processing Systems 2012 (NIPS) (2012) 1097.
16 S. Ioffe and C. Szegedy: 2015 Int. Conf. Machine Learning (ICML) (2015) 448.
17 C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich: 2015 IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (2015) 1.
18 Google tensorflow: https://www.tensorflow.org/ (accessed October 2017).
19 NVidia CUDA: https://developer.nvidia.com/cuda-zone (accessed October 2017).

About the Authors

Chu-Hsing Lin received his Ph.D. degree in computer science from National Tsinghua University, Taiwan, in 1991. He has served as a professor at the Computer Science Department, Tunghai University for over twenty years. Besides teaching and research work, Professor Lin has been the director of the Computer Center, chairman of the Computer Science Department, and director of the Library for Tunghai University. He has also been a Board Member of the Chinese Information Security Association since 2001. Dr. Lin has published over 150 papers in academic journals and international conferences. He has been granted over thirty research projects by government departments and private companies in recent years. In 2006, 2008, and 2010, he was awarded the Outstanding Instructor Award for Master & Ph.D. Theses by the Institute of Information & Computing Machinery, Taiwan. His current research interests include multimedia information security, cryptography, machine learning, and data science.

Jung-Chun Liu received his B.S. degree in Electrical Engineering from National Taiwan University in 1990. He received his M.S. and Ph.D. degrees from the Department of Electrical and Computer Engineering at the University of Texas at Austin, in 1996 and 2004, respectively. He is currently an assistant professor in the Department of Computer Science at Tunghai University, Taiwan. His research interests include cloud computing, embedded systems, big data, network security, artificial intelligence, and sensors.

Ken-Yu Lee received his B.S. degree in computer science from Tunghai University, Taiwan, in 2014. He is currently a Master's program student at Tunghai University. Mr. Lee was awarded the Gold Penguin Award by the Ministry of Economy, Taiwan. His two papers won the first prize and the best paper award in the IEEE ICASI 2017 international conference. His current research interests include machine learning, mobile application, and web service.