

GesID: 3D Gesture Authentication by Deep Autoencoder and Incremental Learning



Xuan WANG

44161514-7

Master of Engineering

Supervisor: Prof. Jiro Tanaka

Graduate School of Information, Production and Systems

Waseda University

July 2018

Abstract

Biometric authentication is widely popular in authentication systems. There are two kinds of biometrics features: one is static physiological characteristics and another is dynamic behavioral characteristics. For static physiological characteristics, if the key information is stolen the authentication will be very insecure. But for dynamic behavioral characteristics, it is very secure because the different individual has different biological characteristics and it is not only hard to steal this information but also hard to behave the information accurately. Gesture as a carrier of behavior characteristics has the advantages of not easily being imitated and containing lots of information. There is a lot of potential value waiting to be excavated in gesture to be a authentication method.

This research aims to use 3D depth information of gesture movement to make authentication. We use a 3D depth camera named Leap Motion as our input device to capture the gesture. First, the system will ask user to behave his free gesture for 3 times at Leap Motion to record these personal movement data, which means the user could customize his personal gesture "password", then a series of preprocessing will be done, after that our system will learn the gesture of the user. In authentication stage, every time when the user wants to pass the authentication, he needs to perform the same gesture. It is worth mentioning that if the user continues to use the system, the system will remember the gesture more secure.

Since authentication is actually an one-class classification (OCC) problem (or called outlier detection and anomaly detection problem), We introduce a deep learning model named autoencoder as our learning method, which takes the high-dimensional gesture data into a one-dimensional mean square error interval. Then a classification boundary strategy is proposed as threshold to make the one-class classification.

Also, in realistic using, asking user to behave too many times of gesture is not a practical solution, but that will cause the data is insufficient at the first training stage. For this difficulty, we employed incremental learning and data augmentation technologies.

Several experiments performed to verify our approach is an effective approach.

Keywords: Gesture, Authentication, One-Class Classification, Autoencoder, Incremental Learning

Acknowledgements

At the beginning, I am very glad to express my sincere gratitude and appreciation to my supervisor Prof.Jiro Tanaka, a very respectable and responsible mentor with rigorous academic attitudes and creative ideas. He has given me lots of help and encouragement, and is always very patient for guiding me to solve problems. Thanks a lot to him for the many valuable and earnest advices during every stage of preparation of this thesis.

Besides, it is grateful to all my dear labmates. They discussed a lot with me and always give me lots of inspiration. Thanks so much for their many helps in experiments so that our experiments could be performed successfully and the good results could be get.

Last but not least, I would like to thank my parents and all love ones. For my parents, they give me financial support and warm mental support throughout my master study life. I will keep going on and making progress to return all those people.

Contents

List of figures	vi
List of tables	viii
1 Introduction	1
1.1 Introduction	1
1.2 Organization of the thesis	3
2 Background	4
2.1 Biometric Authentication	4
2.2 One-Class Classification	5
2.3 Autoencoder	6
2.4 Incremental Learning	7
3 Research Goal and Approach	8
3.1 Goal	8
3.2 Use Case	8
3.3 Approach	10
3.4 Novelty	11
4 System Design	12
4.1 Data Collection	13
4.2 Preprocessing	14
4.2.1 Filtering	14
4.2.2 Data Normalization and 3D Mapping	15
4.2.3 Data Augmentation	18
4.3 One-Class Classification with Autoencoder	19
4.3.1 Structure of Proposed Autoencoder	20
4.4 Threshold Making	21

4.5	Incremental Learning	24
5	System Implementation	26
5.1	Hardware and Data preprocess	26
5.2	Framework	28
5.3	Autoencoder	29
5.4	Network Parameters and Training Process	31
5.5	Classification and Incremental Learning	33
5.6	Graphical User Interface	34
6	Related Work	36
6.1	Related Work about Gesture Authentication	36
6.2	Related Work about One-Class Classification and Autoencoder	37
7	Experiment and Result	39
7.1	Experiment	40
7.2	Result	41
7.2.1	False Rate Analysis	41
7.2.2	Comparison of Simple, Normal and Complicated Gesture	42
7.2.3	Analysis on Data Augmentation	43
7.2.4	Analysis on Incremental Learning	44
7.2.5	Comparison of Previous Works	45
8	Conclusion and Future Work	47
8.1	Conclusion	47
8.2	Future Work	48
	References	49

List of figures

1.1	Different kinds of cheating on static physiological characteristics.	1
1.2	Unlock computer by gesture	2
1.3	Working state of Leap Motion	3
2.1	Compared of there two kinds of classifications	5
2.2	The typical process of an autoencoder	6
3.1	Use case: unlock computer by gesture	9
4.1	Overview of our approach	12
4.2	Hand model in Leap Motion	13
4.3	Data normalization	16
4.4	The result images of data normalization and 3D mapping	18
4.5	The real-time interface of gesture movement	18
4.6	Network structure of autoencoder	21
4.7	The different fluctuation situations	23
4.8	Threshold making schematic diagrams	24
5.1	Leap Motion.	26
5.2	Hand tracking of Leap Motion.	27
5.3	Training Monitor.	29
5.4	Autoencoder structure.	30
5.5	Autoencoder detailed configuration.	31
5.6	The different training stages of 6000 iterations	32
5.7	Different scores of different inputs	33
5.8	Graphical user interface	34
7.1	The gesture authentication system we developed for using and test.	39
7.2	The sample of gestures.	40
7.3	The ROC curve of simple, normal and complicated gesture.	42

7.4	The ROC curve of using or not using data augmentation.	44
7.5	The ROC curve of different incremental learning times (ILT).	45

List of tables

7.1	FAR and FRR score of different gestures after training	42
7.2	FAR and FRR score of using or not using data augmentation (DA).	43
7.3	FAR and FRR score of different incremental learning times (ILT).	45
7.4	Comparison of previous works and our approach.	46

Chapter 1

Introduction

1.1 Introduction

In recent authentication researches, biometric authentication has increasingly become a research hotspot. The biometric authentication can be further divided into static physiological characteristics and dynamic behavioral characteristics. Static physiological characteristics refer to the technology of personal identification by using the physical characteristics that are inherent in the human body. In static physiological characteristics, some applications [1] such as Apple face ID has been widely used in the market. Although static physiological characteristics have made some progress, their weakness that easily be imitated still exist [2]. There are cases shown in Fig. 1.1, which are some cases that fingerprint copy for cheating and imitating handwriting.

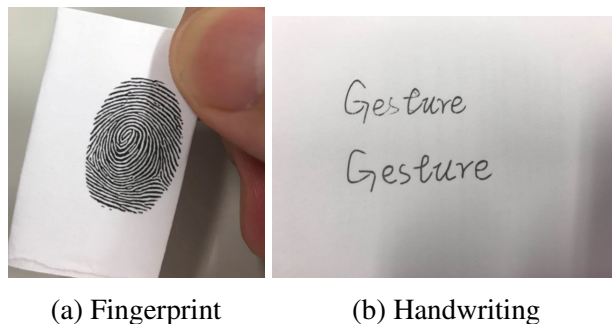


Fig. 1.1 Different kinds of cheating on static physiological characteristics. (a) is fingerprint copy and (b) is imitating handwriting.

As for the dynamic behavioral features, because different users have different habits, different muscle memories [3], their behavior habits are very difficult to be told by intuitive vision or experience, so these kinds of habits are not easily copied and imitated. As a kind of dynamic behavior characteristics, gesture has these three advantages [4]: easy to express, large information content and difficult to be imitated. Therefore, it is very suitable to be a carrier of authentication. The movement of gesture itself can fully represent the of users themselves [5], so we aim to study the possible methods and possibility of gesture based biometric authentication. The main purpose of this study is also to explore an effective and robust 3D gesture authentication solution.

Consider the following use case shown in Fig. 1.2, a user wants to unlock his computer, and he just needs to behave his own gesture "password" in front of his computer and the computer will be awoken by his gesture. This whole process is very natural and smooth, and for this user, the gesture is also very easy to be remembered. Of course, just like we need to set our text password first as registering, the system will also ask user to register his gesture "password" and remember the "password" in registering stage.

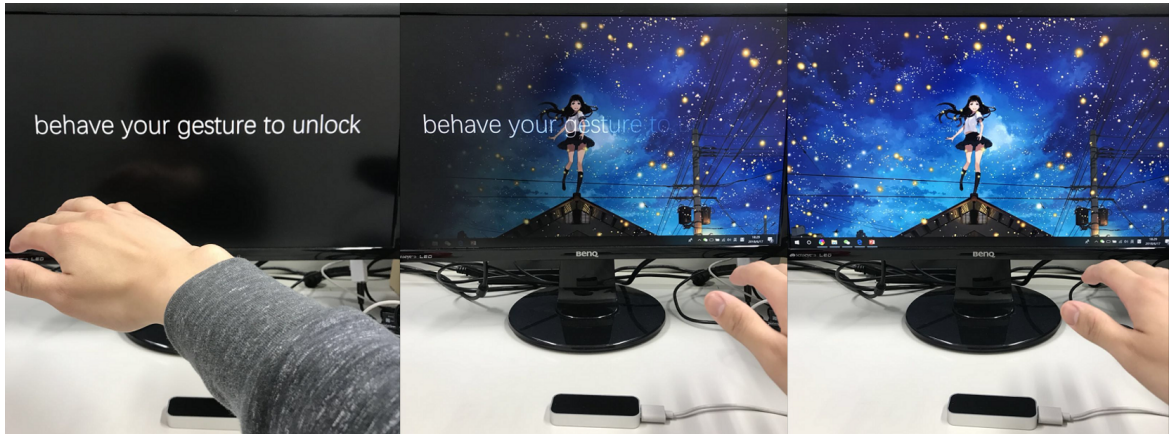


Fig. 1.2 Unlock computer by gesture

In this study, we use a depth camera named Leap Motion as data input device shown in Fig. 1.3, which is an infrared-based depth camera that can be used for tracking hands and fingers. For registering gesture, the user needs to behave in front of Leap Motion several times. After learning the gesture model, the user could just behave the same gesture to unlock the system.



Fig. 1.3 Working state of Leap Motion

Besides, in order to make the system learn user's gesture more accurately, the system will learn the gesture more while using. This learning mechanism is very similar to human, because as the knowledge is updating, the system needs not only learning the new knowledge but also learning the common knowledge more and more firmly.

1.2 Organization of the thesis

The rest of the thesis is organized as follows: Chapter 2 introduces the background about the thesis and also the related works in this field. Chapter 3 will tell the research goal and also the approach will be told briefly. Chapter 4 is the system design part, where the design concept and ideas will be introduced and the algorithm design will also be told. Chapter 5 will be the system implementation part where the detailed environment and implementation will be talked. Chapter 6 will introduce the related work. Chapter 7 will be about the experiments, we will talk about the performance of our approach and the comparison of different approached will also be done. The last part, Chapter 8, will be conclusion and future work part, where we will conclude the previous content and talk about the future possibilities.

Chapter 2

Background

2.1 Biometric Authentication

Biometrics technology mainly refers to a technology for identity authentication through human biometrics. Human biometrics are often unique, can be measured or can be automatically identified and validated, genetic and lifelong, so biometric authentication technology has greater advantages than traditional identification techniques.

The biometric system samples biological features, extracts and compresses their unique features and converts them into information of codes, and further converts these codes into features' templates. Biometrics include not only physiological static characteristics like fingerprint, face, ear shape, retina, iris, pulse, and so on, but also dynamic behavior characteristics like voice, and gesture. Based on these features, people have developed various biometric technologies such as hand recognition, fingerprint recognition, face recognition, speech recognition, iris recognition, signature recognition, and many other applications.

Since the human body has characteristics unique to the others, the characteristics of living features could not be copied, stolen, or forgotten. It is safe, reliable, and accurate to use the kind of authentication methods of the characteristics of living feature. Compared to traditional methods which are very easy to be lost, forgotten, copied and stolen like password, IC card, magnetic card, and key. You do not need to have a key holding a big bunch of keys

and you also do not have to remember the password. Under the modern computer technology, these products are easy to be realized by computer.

2.2 One-Class Classification

When classification is mentioned, multi-class classification is the concept occurred in our mind, which means the task that classify different classes with a certain label. For example, there are many kinds of birds in the world, and you can classify the what bird is by your eyes. For machine, it can also classify the bird by "learning" it, which is a process that you must input the basic knowledge to it and it will gradually learn that.

But here we will introduce a concept of One-Class Classification (OCC). Compared to multi-class classification, OCC is the classification that only one class input dataset is provided but the task needs to classify this class and others. So in some ways, OCC is also called as outlier detection or anomaly detection.

A few of researchers have done some discussions on it, and there are a number of algorithms that can be used [6]. Manevitz et al. mentioned the OCC solution based on autoencoder [7], and they gave a discussion of the performance between autoencoder and other methods.

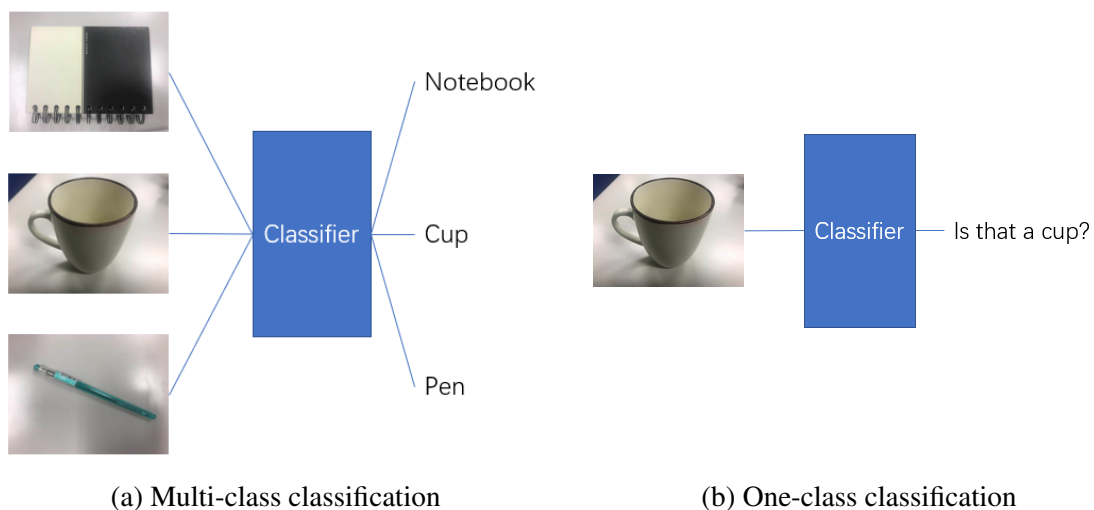


Fig. 2.1 Compared of there two kinds of classifications

Authentication is this process: to judge the person is authorized or not, and what we got is that the authorized information. So this problem is a OCC problem in some ways, where the authorized information is the positive class and you need to classify this class and other classes.

2.3 Autoencoder

The concept of deep learning is from the research of artificial neural network. Multilayer perceptron with multiple hidden layers is a deep learning structure. Deep learning forms a more abstract high-level representation of attribute classes or features by combining low-level features to discover distributed feature representations of data.

Autoencoder is a kind of neural network within the concept of deep learning. Usually autoencoder has an input layer, a hidden layer and an output layer. Autoencoder learns to compress data from the input uncompress it into something that closely matches (or fits) the original data.

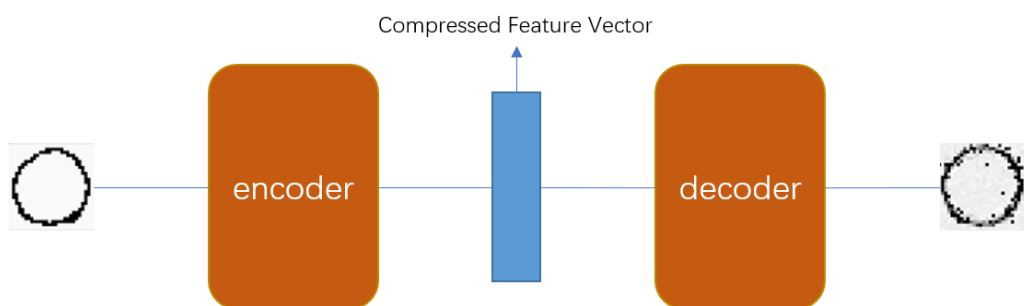


Fig. 2.2 The typical process of an autoencoder

The autoencoder consists of two parts:

- Encoder: The encoder will compress input data into latent space representation, and can be represented by encoding function $h=f(x)$.
- Decoder: this decoder will reconstruct the input from the potential spatial representation, and can be represented by the decoding function $r=g(H)$.

2.4 Incremental Learning

Incremental learning refers to the ability of a learning system to learn new knowledge from new samples and to preserve most of the knowledge that has been learned before. Incremental learning is very similar to human's own learning mode. For people to learn and receive new things every day during their growing up, learning is progressively carried out, and human beings are generally not forgotten about the knowledge that has been learned.

Compared to Incremental learning, batch learning is well-known mode for most of machine learning system. Because in most situations, the size of dataset is certain and the system just need to learn this dataset. But in some situations that the knowledge is increased by time, there are always new knowledge generates, so it is very important for a system to learn that new knowledges.

Incremental learning is very suitable for authentication system. Because at first in training stage people cannot be asked for behaving too many times of gestures, as an learning strategy, keeping the machine learning while the positive class comes will increase the accuracy of the learning system.

Chapter 3

Research Goal and Approach

3.1 Goal

The fundamental goal of our research is to **find a robust gesture authentication solution with less effort.**

Previous works using dynamic characteristic like gesture as authentication method need a lot of thing to be done: find many users and collect their gesture information by a lot of behaving. This is not practical in realistic using, we want to minimize these works and at the same time ensure the high accuracy, so our system should also satisfy these 3 requirements:

1. Could be used by just single user's customized gesture;
2. Could be used without too many times of behaving gesture but it will still have high accuracy;
3. Could be used to continue learning user's gesture while user is using.

3.2 Use Case

Assume the following situation: 1 user wants to pass the authentication by his gesture.

Before this use case, the system has already learned the user's gesture: the user behaved his gesture 3 times in registering stage and the system learned his gesture.

The precondition is: the system is listening to user's gesture.

The operation process is: this user puts his hand on the top of Leap Motion and tries to behave his gesture according to his memory. After a few seconds processing, the animation of result will be displayed on the screen.

There are two possible results: accept and reject. In the accept situation, the system will let the user pass the authentication and at the same time, this new gesture will be learned incrementally, but this process user will not see; In the reject situation, the system will not let the user pass and ask the user try again.

The use case above will be shown in Fig. 3.1. At first the system is listening, if the gesture is accepted, the animation of success will be shown, followed by the welcome interface; If the gesture is rejected, "please try again" will be shown on the screen.



Fig. 3.1 Use case: unlock computer by gesture

3.3 Approach

To achieve the goal, we will introduce our approach of on the following aspects. Firstly, 3 key aspects will be introduced: one-class classification with autoencoder, data augmentation and incremental learning.

In order to make classification by one single user and of high accuracy, we would like to regard gesture authentication as an "one-class classification" (OCC) [8] problem. Due to the particularity of the authentication, which is mostly for the user's single use scene. Therefore, authentication can be considered as anomaly detection, and the anomaly detection is an OCC problem. Unlike the previous studies [9–13] in gesture authentication, the system based on OCC could be more called "authentication", because it can directly identify the user itself without the need to rely on other user groups. Gesture authentication is mostly multi-class classification in the previous studies, which means the system can only distinguish which user is the "authorized user", but it cannot directly identify the user without others' information. Also, the requirement of multi-class classification can be realized when multiple OCC systems are assembled together. We use autoencoder as our OCC model. The autoencoder we employed is a 5 layers neural network. It will compress the input data into a feature vector and then uncompress the feature vector into original size, then a loss function named mean squared error will be used to evaluate the difference from input and output as a labeling process. In other words, autoencoder actually learns user's positive gesture in training stage, and in test stage, it will map the high-dimensional gesture data to one-dimensional mean square error interval. For classifying true and false classes, a boundary making strategy is proposed by us to judge if the test data is in positive interval or negative interval.

In order to overcome the difficulty that deep learning requires a large number of datasets and to ensure the accuracy of the results, we use data augmentation technology. Data augmentation [14] is a very effective solution in the case of insufficient data, and the amount of data is added to the workable level by the specific operations of the data.

In order to let the system continue learning user's gesture while user is using, we use incremental learning technology. The incremental learning [15] is a human-like learning mechanism of machine learning, which will know the new knowledge without forgetting old

knowledge, just like human's learning mechanism. Incremental learning will increase the accuracy of the system by re-learning the model while using.

Then here are the other aspects of our research approach for establishing the robust system.

In order to capture 3D dynamic gesture movement, we employ Leap Motion as our data input device. Because depth camera is gradually popular, more and more depth camera will be used in our daily life. This will make depth camera based authentication has high usability.

In order to make the Leap Motion more robust for the system, we use a series of data preprocessing, including filtering and data normalization to achieve more effective and robust algorithm. These kinds of methods are mature in many situation's usages. Due to the problem that Leap Motion will produce noise and the data position is not in the center, we much enhance processing procedure to improve the result, we believe these kinds of process are very important in both study propose and business propose.

To verify our approach achieves our goal and prove the effectiveness of our approach, we will perform several experiments.

3.4 Novelty

The novelty of our research mainly reflects in these aspects:

1. We propose using one-class classification with autoencoder as learning method in gesture authentication fields;
2. The boundary making strategy is new for doing one-class classification;
3. To minimize the effort of user. Data augmentation is used for solving the problem of insufficient data amount when training. Also, incremental learning technology allows the system continue learning user's gesture while using.

Chapter 4

System Design

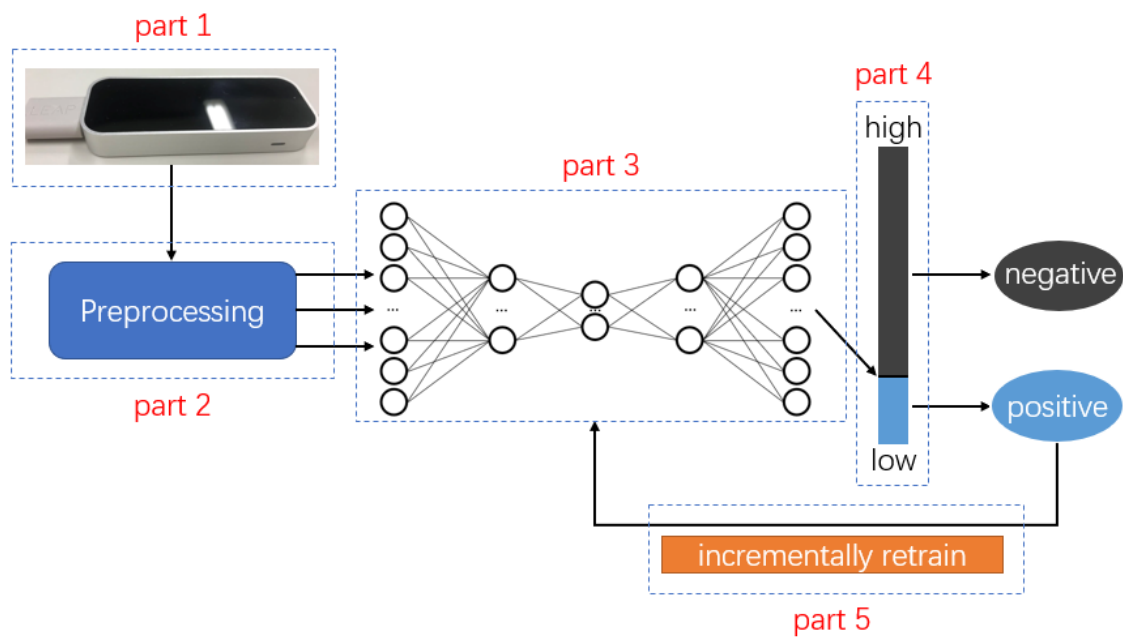


Fig. 4.1 Overview of our approach

In this chapter, we will introduce our system design and each point of our approach. The Fig. 4.1 shows the overall structure of the algorithm, and we will divide the algorithm description into five parts:

- Part 1 is about the data collection from Leap Motion device;

- Part 2 is the preprocessing part. Filtering, data normalization and 3D mapping will be introduced;
- Part 3 introduces autoencoder's structure we want to use, and how our autoencoder works.
- Part 4 proposes a new threshold making strategy;
- part 5 is about the incremental learning.

What we would like to emphasize will be Part 3, 4 and 5.

4.1 Data Collection

In our proposed system, we select Leap Motion as our data source to collect gesture raw data. Fig. 4.2 shows the captured image of its viewer. Leap Motion is a very small device that has different types of infrared-based depth camera inside, which is specially designed for tracking and detecting human hand movement. The data from Leap Motion contains lots of information such as the location and velocity in virtual coordinate system.

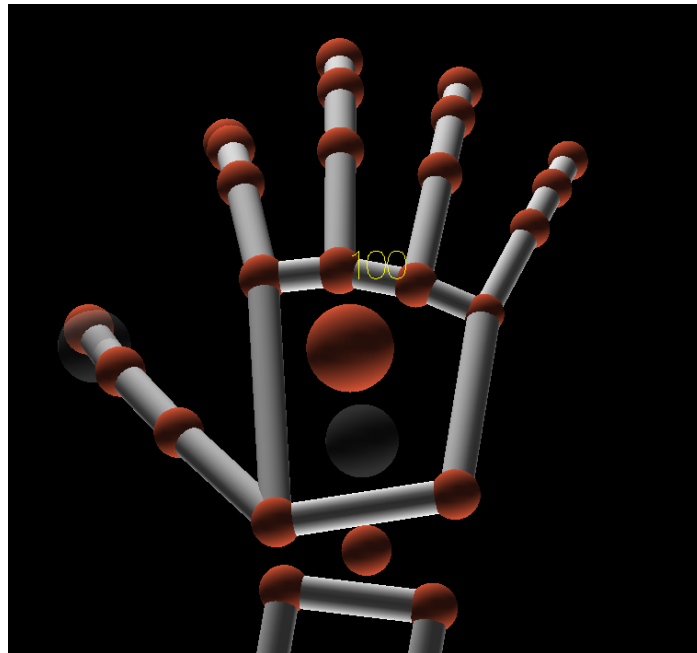


Fig. 4.2 Hand model in Leap Motion

For gesture information collection, we have considered several devices as input device, just as the devices that previous researches [9–13] employed. There are mainly two reasons for choosing this device. In the first place, the depth camera has been applied to the market in recent years, which has a promising future in a very large number of fields; secondly, this device is very suitable for the movement tracking of hand and has made a lot of optimization.

For deciding what kind of data that we want to get from gestures for further processing, we did some investigation [5]. In the generalized concept of gesture, there are several kinds of gesture besides hand gesture, such as arm gesture which can also be called gestures. But we found that the moving state of hand is a key component of gesture, so we decide to get the data on fingertip of five fingers from the device, including the position and the current velocity information.

4.2 Preprocessing

4.2.1 Filtering

Since filtering technology is a very mature technology, we will introduce the filter very briefly, just in order to make our preprocessing work clearer. In the early tests, we found that there were two problems caused by the device in the data transmission process:

1. The data flow was not stable, and the isolated point (salt and pepper noise) would be generated from time to time;
2. The data is not smooth, instead it will produce fluctuations.

For the reason that gesture authentication does not require very detailed data points, we decide to use fuzzy filter to process the data. Here, median filter and Gauss filter are used to solve these two problems respectively. First, the median filter will be used. The dataset from the depth camera is a three-dimensional point cloud dataset, so three-dimensional median filter with window size of 5 will be used, and the filtering process is defined by the following formula:

$$g[x, y, z] = \text{med}\{f[x, y, z], (x, y, z) \in W\} \quad (4.1)$$

where W is the set of front and rear five data points of the current data as the sliding window; $g[x, y, z]$ represents the three-dimensional coordinates of a filtered certain point while $f[x, y, z]$ represents the data before filtering. As a result, the outliers will be filtered out, so that there will be no mutational point in the data which has an obvious effect on the model's classification result. Since median filtering is a non-linear filter, the unsmooth characteristics of the data still exist. Here we use the Gauss filter with filter kernel size of $5*5*5$, $\sigma=0.8$, and then perform the three-dimensional convolution operation. Here, the values of each point in the kernel will be determined by the following Gauss distribution:

$$G(x, y, z) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2+z^2}{2\sigma^2}} \quad (4.2)$$

where σ is standard deviation of whole data. This formula is the principle of Gaussian kernel's generation. After obtaining the Gaussian kernel, we can perform three-dimensional convolution operation with it on the data.

4.2.2 Data Normalization and 3D Mapping

Since the dataset we get is all three dimensions, in other words one gesture is characterized by three features, and each gesture will be measured by establishing a three-dimensional coordinate system. In addition, the position relative to device is usually different in every time of gesture behaving, the coordinates of movement track are also different. If these different coordinates are established in the same coordinate system, errors will be produced during the subsequent learning process. Even in the current study, neural networks have been able to be size-insensitive. The size-sensitive data input will have better fitting results, which is more accurate than that without data normalization. In such sense we do normalization operation on the data after filtering procedure. Its purpose is to eliminate irrelevant differences among each gesture so that our model can learn the key data. Also, for the importance of normalization has been discussed by previous research [16].

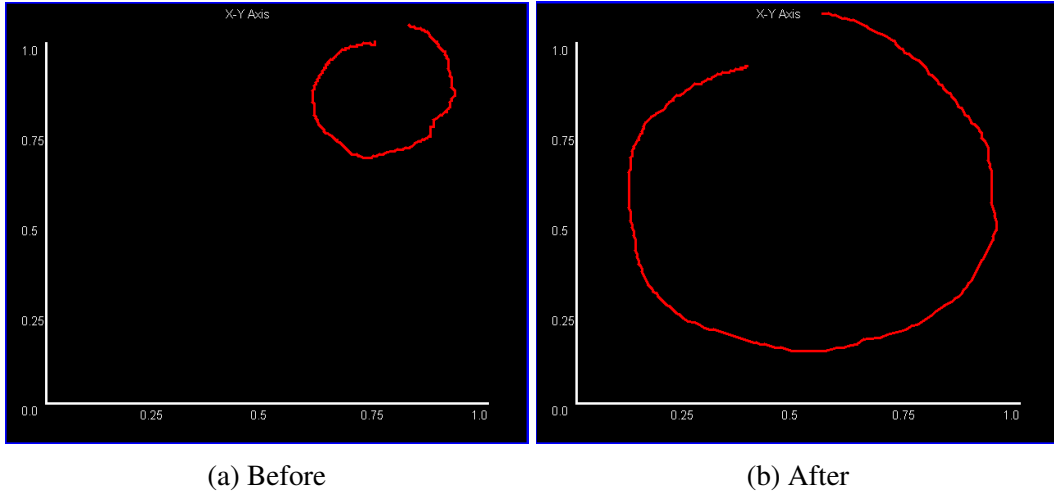


Fig. 4.3 This figure shows similar gesture (“circle” gesture mapped in X-Y plane) but in different position relative to device, where the red line is the mapping of 3D gesture. (a) This gesture is farther and more biased from device. (b) This gesture is just behaved on the center and top of device.

Fig. 4.3 shows this difference in similar gesture (for explanation, in this figure gesture movement is just mapped to X-Y plane) on different relative position to device. If there is no normalization operation the “congenital” input error will exist, it is very hard for a classifier to judge if there two gestures are similar or not.

Here our normalization method is to scale the gesture data in proportion to a small specific interval, and to remove the unit limit of the data and convert it into a pure value. In order to make the next step of classification more accurate and easy to operate, we also map the original three-dimensional data into three two-dimensional data. We are using a “window scanner” to scan this three-dimensional movement and produce the result two-dimensional data, which can also be regard as an image. For the reason that making classification step more accurate and easy to operate, we define the size of mapping result as 28×28 and every point of this result is 0 or 1, which means 784 data points will be used for the next classification.

Our algorithm in normalization is performed by the following steps:

- (1) Generate 676 ($=26*26$) two-dimensional scanning windows, the start location of window can be described as follows (here we take the mapping of X-Y plane as an example, X-Z, Y-Z plane is in the same way):

$$w_{i,j} = [26i(x_{max} - x_{min}), 26j(y_{max} - y_{min})] \quad (4.3)$$

where $w_{i,j}$ is start location of scanning window, x_{max} is the max value in X dimension of gesture, x_{min} , y_{max} and y_{min} are similar. In fact, the reason we used the window size $26*26$ instead of $28*28$ is to prepare for the next data augmentation. See the data augmentation below for details;

- (2) For each window, it will scan the certain plane (e.g. X-Y plane's scanning means only X dimension's data and Y dimension's data will be used) of gesture and if there exists at least one data, the mapped result will be 1 otherwise it will be 0, also can be represented as follows:

$$r_{i,j} = \begin{cases} 0, & \text{there exists positive value in } w_{i,j} \\ 1, & \text{there are all 0 in } w_{i,j} \end{cases} \quad (4.4)$$

where $r_{i,j}$ means the point in result matrix;

- (3) Generate four edges of the result, adding a row (or column) of all 0 value to the top, bottom, left, and right.
- (4) Repeat (1) ~ (3) until the three planes are all scanned.

In order to observe the results of this series of processes more intuitively, Fig. 4.4 shows the example of "circle" gesture of three different planes after processing (we generate images for observe the result, but it looks very small because the size is $28*28$).

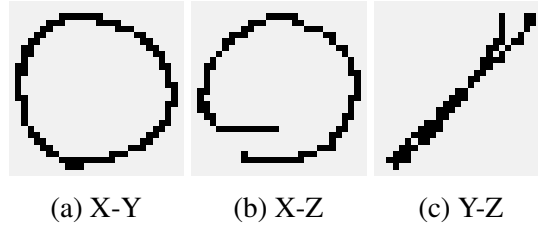


Fig. 4.4 The result images of data normalization and 3D mapping. (a), (b) and (c) is the mapping of X-Y, X-Z and Y-Z plane respectively. For the reason that gesture “circle” is three-dimensional, in X-Z and Y-Z plane they may not looks like a “circle”.

For convenience, we also made an interface to facilitate observation and operation. Fig. 4.5 is the interface of three planes when users are using. Fig. 4.4 shows the result image file and Fig. 5 is the real-time interface when operating, the gesture in Fig. 4.5 is also different from Fig. 4.4.

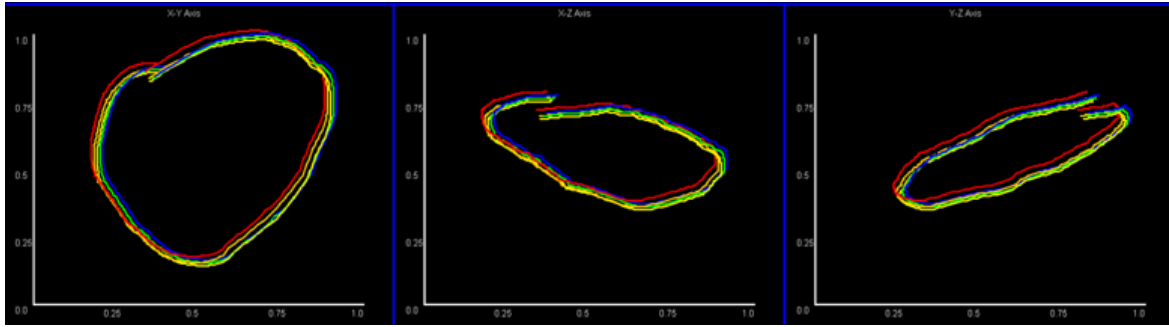


Fig. 4.5 The real-time interface of gesture movement. The data has been normalized and these 5 colors means 5 fingers’ movement. From left to right X-Y, X-Z and Y-Z plane’s gesture mapping is showed separately.

4.2.3 Data Augmentation

In the consideration of practical algorithm design, we find that it is a very unrealistic way for users to behave gesture too many times. But usually deep learning needs a lot of data to train the network. So based on a previous research [17], we use the method of data augmentation to expand the dataset. Another advantage of data augmentation is that it could reduce the overfitting phenomenon of the network. Through the transformation of the training pictures, we can get a better generalization ability of the network, and better applicability to the application scene.

In order to not change the size and shape, we use shift, flip and rotation as data augmentation operations, instead we don't choose operations like room in and out which could change the image's shape or size. Under these operations each original data could become 180 data, which greatly expands the scale of the dataset.

4.3 One-Class Classification with Autoencoder

When the classification problem is mentioned, most of the cases are described below: there are two or more classes of data to be trained, which will be put into a classifier model with their labels to learn, and when finished the model can be used for doing more classification to classify them to a certain label.

But consider the following situation: there is a problem, in the training process there will be only positive data and no other data as a contrast. In other words, there is only one label. At this time the key problem is that when the model is trained to use, there will be negative data coming with using, and the model needs to classify negative data from positive dataset. However, there are no samples of negative data in the training process, and negative data themselves are varied and changeable. This kind of problem is called one-class classification (OCC) as mentioned in the introduction part. The problem is also named as anomaly detection or outlier detection. The key is to draw the boundaries of positive data. Any data falling within the boundary will be considered as positive data, and all data outside the boundary will be considered as negative data.

In the case of gesture authentication, we think that all data are positive data during the training process, and there would be negative data in using, so it is a very typical OCC problem. Moreover, how to clearly classify user data and non-user data in this problem is a key issue.

From a survey [6] of previous researches, the current solutions are mainly divided into one-class support vector machine (OSVM) and other methods (hidden Markov model, nearest neighbor and so on). These methods play a very important role in the development

of one-class classification, the detailed comparison will be given in the Chapter 6.5. Our method is using autoencoder (AE) as classifier, and the boundary is made by certain strategy.

4.3.1 Structure of Proposed Autoencoder

Autoencoder [18] is an unsupervised feed-forward neural network. It is divided into encoder and decoder. The input data is compressed to hidden layer and then decompressed to its original size. For the reason that the compression is lossy, autoencoder can approximatively copy the input, then the useful features of data will be learned by autoencoders. Autoencoders are data related, if an autoencoder has learned a class of features, then compression of this class will have a good result, but the performance of compression of other classes will be very poor. The encoding process can be represented as follows:

$$\mathbf{h} = f_{enc}(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (4.5)$$

where $\mathbf{h} \in \mathbf{R}$ means the hidden feature vector of autoencoder and $\mathbf{x} \in \mathbf{R}$ is the input data matrix, \mathbf{W} is a weight matrix and \mathbf{b} is the bias vector of encoding process. Autoencoder will also decode this feature vector:

$$\mathbf{x}' = f_{dec}(\mathbf{W}'\mathbf{h} + \mathbf{b}') \quad (4.6)$$

where \mathbf{x}' is final output of autoencoder, \mathbf{W}' and \mathbf{b}' are the parameters in decoding process. \mathbf{x}' and \mathbf{x} have the similarity and what autoencoder do is to minimize the difference between them.

The learning process of autoencoder is to compare the input and output, and to train the autoencoder according to the loss by the stochastic gradient descent algorithm. Our loss function will be:

$$L_{MSE}(\mathbf{x}, \mathbf{x}') = \frac{1}{n} \sum_{i=1}^N \|\mathbf{x} - \mathbf{x}'\| \quad (4.7)$$

where L_{MSE} is the loss function that is used to punish the difference between input and output with n predictions, and there are several functions can be used as loss function, in

our case, we select mean squared error (MSE) as the loss function. Because our input and output are two-dimensional data which can be regarded as images, MSE score is an effective measurement.

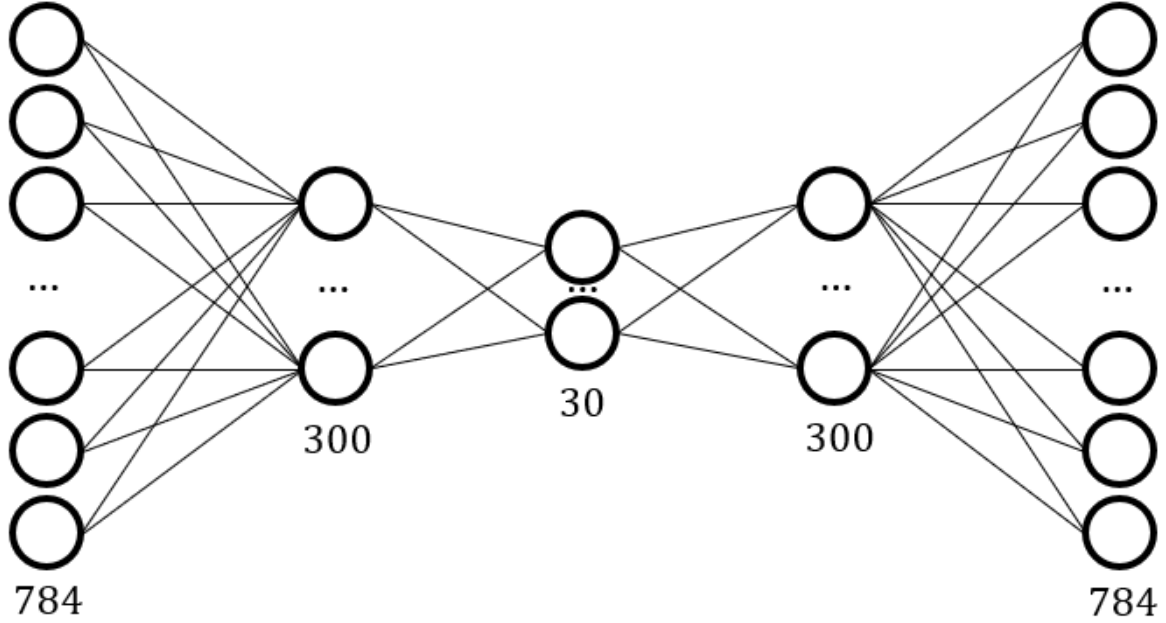


Fig. 4.6 Network structure of autoencoder

Fig. 4.6 shows the structure of our autoencoder, this network structure is based on the results of many experiments that we can choose to balance the best match reduction degree and specificity matching.

4.4 Threshold Making

So far, our algorithm has been able to map the input of gestures to a one-dimensional linear interval, but how to classify such a one-dimensional linear space is still a problem to be explored. Within our knowledge, we have found some previous studies. Nathalie Japkowicz et al. [18] firstly proposed a threshold determination approach that relax the last score of training result by 25% higher than the score. Larry Manevitz et al. [7] optimized that idea but tighten the training result and proposed another strategy that using the score at 90th percentile, because in that stage the training score usually is not the best score, in other

words the score at 90th percentile is a little higher than the last, which can be regarded as the boundary. These ideas are very enlightening, based on the study of these researches we propose our strategy of boundary making.

During training the network, we find that the training curve has fluctuating characteristics. According to different gestures, the degree of fluctuation is different. The key to our border making strategy is to choose the boundary loose or tight according to the fluctuation of the data: the boundary should have better capacity when the fluctuation of training data is relatively large; while the boundary will be more contracted when the training data is relatively stable. In addition, in the process of the training of the autoencoder, MSE Score is gradually converging with the training, in other words, there will be a gradual optimizing trend, so the score of the last part of the training will produce better results as the reference data. To sum up, we propose the following strategy of boundary making:

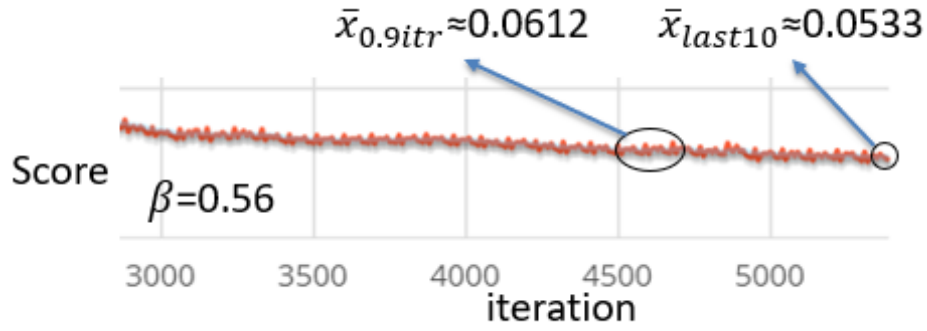
$$T = (1 + \beta^2) \frac{T_{0.9itr} * T_{last10}}{(\beta^2 * T_{0.9itr}) + T_{last10}} \quad (4.8)$$

and

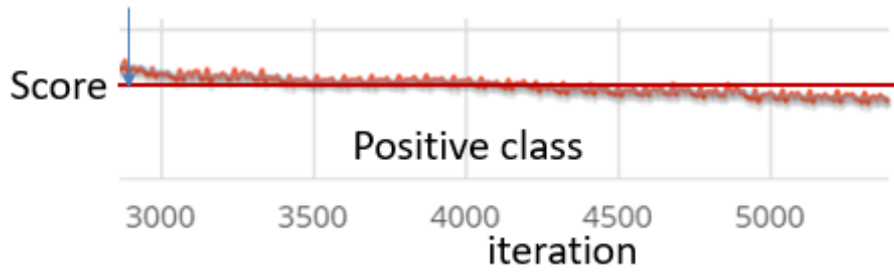
$$\beta = \sum_{i=0.6itr}^n |x_i - x_{i-1}| \quad (4.9)$$

$$T_{0.9itr} = \overline{x_{0.9itr}}, T_{last10} = s\overline{x_{last10}} \quad (4.10)$$

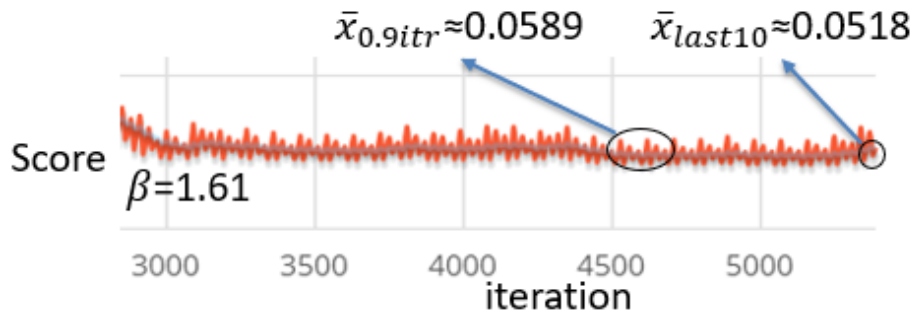
where T is the threshold, β is proportion of each part and represents the fluctuation degree, s is relaxing parameter (set as 1.15, which means it is relaxed by 15%), $\overline{x_{last10}}$ is the last 10 iterations' mean MSE value and $\overline{x_{0.9itr}}$ is the mean MSE value of 50 iterations located at 90th percentile. This strategy is to balance the $T_{0.9itr}$ and T_{last10} by the proportion parameter β and as a statistic, we made fifty experiments and found that the value is between $0.35 \sim 1.73$, depends on the different fluctuation situations. Fig. 4.7 shows different fluctuation situations and the boundary situations using our boundary making strategy.



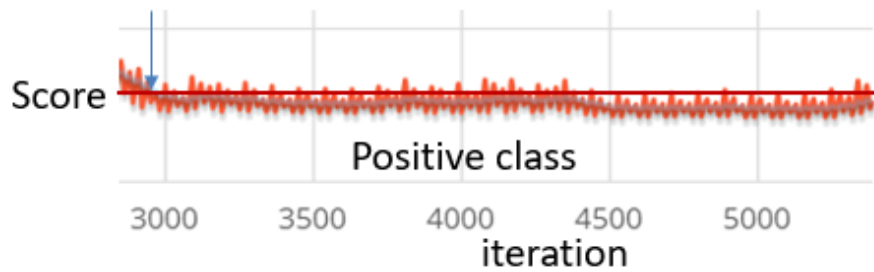
boundary $T = 0.06122265288992104$



(a) stable situation



boundary $T = 0.05938195336382738$



(b) unstable situation

Fig. 4.7 The different fluctuation situations, and boundary situations, the curve is the intercepted segment of last part of training. (a) is the a relative stable situation while (b) is relative unstable. T is the boundary shown by a red line.

The main reason for this strategy is to avoid the problem that happen when any of the two strategies is used alone. If only the final results (T_{last10}) are used, when the final segment just shrink to the lowest point but only the local lowest point, the boundary setting does not have good classification characteristics. But if only the 90th percentile's mean score ($T_{0.9itr}$) are used, when the final segment is very smooth and hardly to see any optimization, the score located at 90th percentile is almost the same as the last score, which also does not have good classification characteristics. We drew two schematic diagrams to represent the two problems respectively at Fig. 4.8.

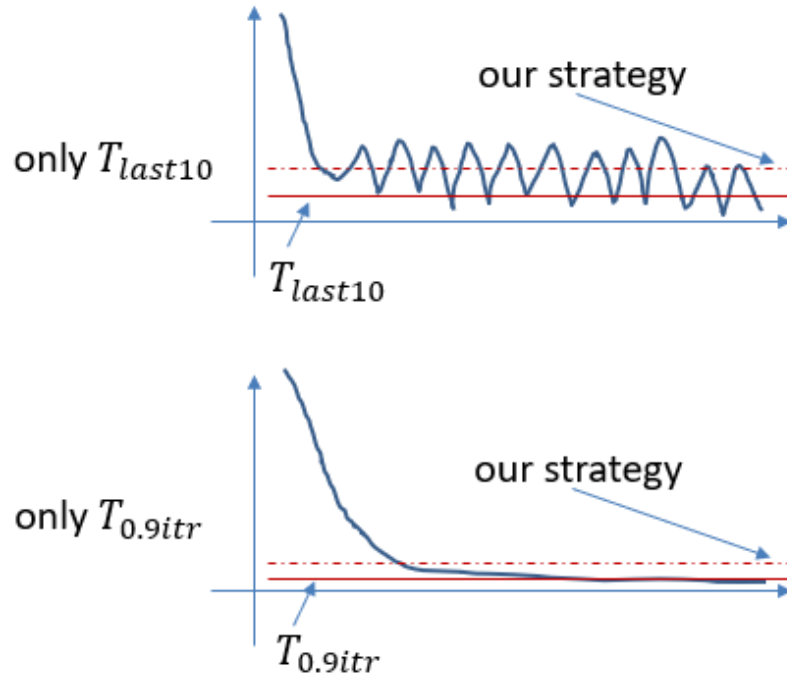


Fig. 4.8 Schematic diagrams to explain the problems when only one threshold is used.

4.5 Incremental Learning

If a learning system has the ability that it can gradually learn the new knowledge from new batch of dataset and preserve knowledge that has already learned before, we will call that this system has the ability of incremental learning. This learning mode is more like a

human learning mode, because it is just like the human who learn and receive knowledge everyday and most of memory will not be forgotten.

In our design, our approach will gradually learn the new batch while using. Assume the user behaves his gesture and pass the authentication, this batch of gesture will be re-trained to fit the autoencoder and generate result as the direction of gradient decent.

There are two main reasons of using incremental learning: one reason is our of user-friendly consideration that it is not practical for user to behave too many times of gesture at training stage, so using incremental learning will greatly solve this problem because it will let not behaving too many times of gesture be possible; Another reason is that incremental learning method will also increase the accuracy while lots of using, because the autoencoder gradually learns the user's habit, the unique points from others, that will largely reduce the false rejection rate of the system.

The evaluation of incremental learning will be given on the experiments part.

Chapter 5

System Implementation

5.1 Hardware and Data preprocess

Leap motion is an infrared based depth camera hardware with very small volume. As shown in the following Fig. 5.1.



Fig. 5.1 Leap Motion.

The two internal infrared sensors will capture the position and speed vector of the hand through different angles, and ensure the accuracy and accuracy of hand tracking through their rich algorithms. Fig. 5.2 shows the hand tracking of Leap Motion.

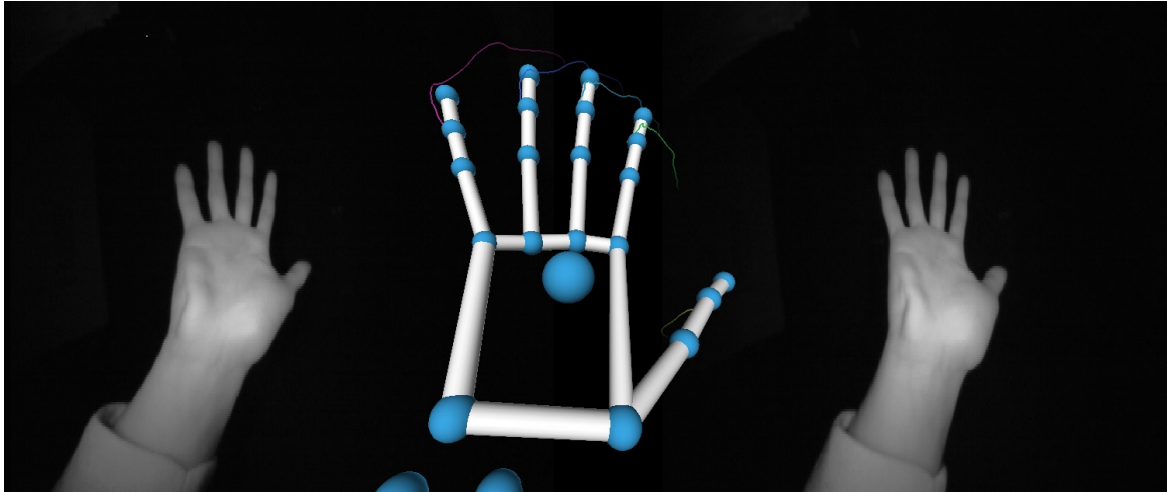


Fig. 5.2 Hand tracking of Leap Motion.

Leap Motion provides a series of APIs for developers, this API contains your hand model in different classes. This information includes your hand's position, velocity, acceleration of your every finger, palm and even every bone in every frame.

For our gesture tracking capture, we need to both get the position and velocity information for gesture collection. The information is in the class `Fingers`, where the vector of position and velocity are both exist.

In every frame, Leap Motion will call "`onFrame()`" to do the operation defined by programmer. The `Frame` class contains current hand gesture information, and we will save these data in array and put the data in buffer are to use in later stage.

Because data preprocessing is not the key part of our system, here we just briefly introduce it.

The buffer has a fixed length to save a period of data, and basing that data buffer we will use our filters to smooth the data in buffer. Leap Motion is a very precise device for collecting hand information but sometimes light or other problems will cause some noises.

For authentication is a very reliable data processing, we need to smooth the noises. So now after filtering the buffer we have got the smoothed buffer.

The next step is data normalization and 3D mapping. Also, as mentioned in Chapter 4, a window will be create, whose width is equal to total width of gesture divided by output width, and height also the same. The window will scan the gesture and get the output images.

5.2 Framework

Deep learning is a gradually growing field of computer science. From the appearance of deep learning to the current stage, more and more projects have been developed and applied, and the essence of deep learning is also gradually revealed. In deep learning, writing network structure from layer to layer requires most repetitive work and time. In such sense, deep learning framework has been created which is a tool to write deep learning projects for the efficiency, completeness and rapidity.

We choose Deeplearning4j [19] as our framework to write our neural network. This framework is established for JVM environment providing a set of solutions for deep learning, which contains mathematic library, neural network configuration builder, activation function library, loss function library, training strategy and so on.

When we configure the network and specify the data source for the framework, it will automatically call the resources of the system to customize the training of your neural network, and will give a set of tools to monitor the training of the neural network in real time. The training monitor will be shown in Fig. 5.3. The framework provides great convenience for our work.

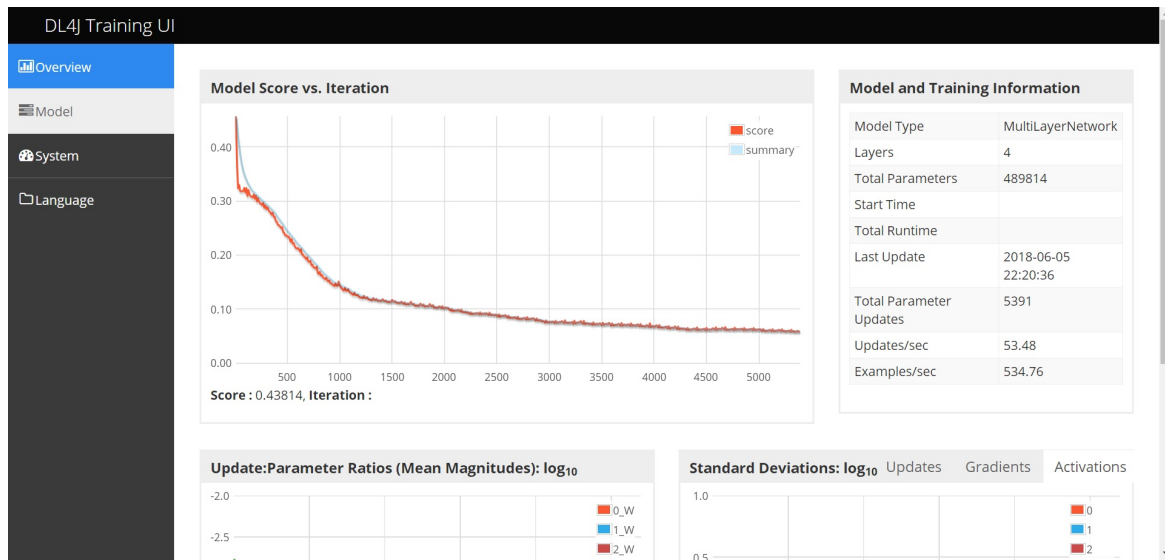


Fig. 5.3 Training Monitor.

5.3 Autoencoder

Based on the framework mentioned in Section 5.2, we construct the autoencoder according to the principle in Chapter 4. As shown in Fig. 5.4, the input image will be put in the autoencoder, after that encoding process (From 784 to 30) and decoding process (From 30 to 784) will be performed as a lossy compression and decompression process. The result will be compared with the input image for similarity, the similarity standard is mean squared error (MSE) between them. After calculating the MSE loss, output layer's weights and bias will be adjusted to the gradient decent direction, and then every layer's weights and bias will be all adjusted by back propagation algorithm.

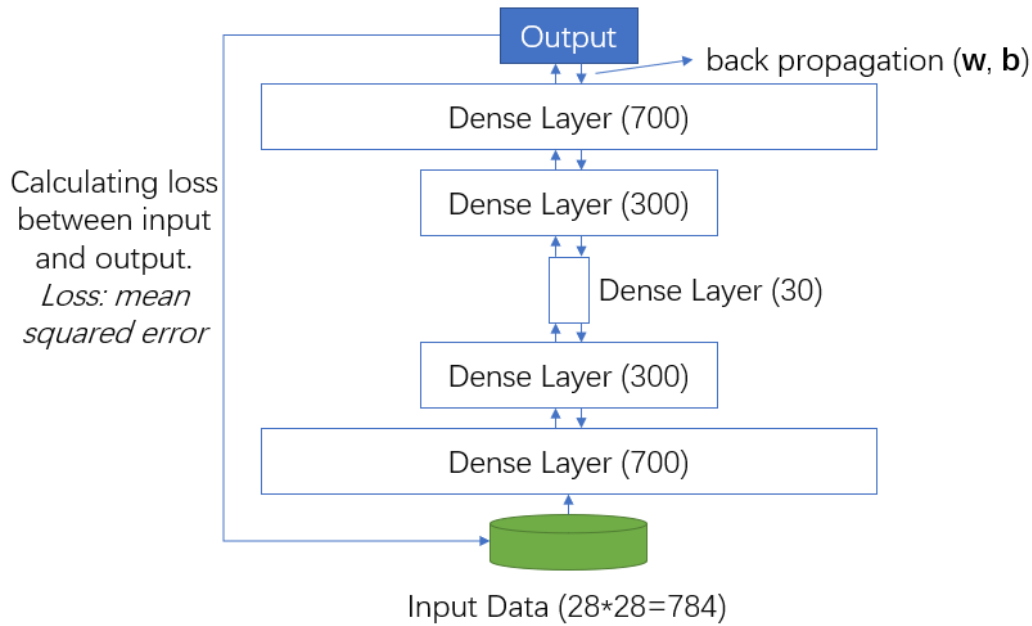


Fig. 5.4 Autoencoder structure.

The whole network is an unsupervised neural network where the input data itself could be the label of its output. The bigger their difference is, the bigger the loss will be. The degree of similarity can be used as a criterion for judging abnormal detection. Since the autoencoder has been trained to the exclusive compression and decompression process, once the nonstandard data is put into the autoencoder, the output results will not be guaranteed. Therefore, the MSE score is used to compare the input and output, and we can effectively draw the gap between the input and output, and this gap is an effective basis for our classification.

As an intuitive impression, Section 5.4 will tell how the difference looks when a positive data comes and a negative data comes. For more detail, Fig. 5.5 gives the detailed configuration of our autoencoder.

```

MultiLayerConfiguration conf = new NeuralNetConfiguration.Builder()
    .seed(123456)
    .optimizationAlgo( OptimizationAlgorithm.STOCHASTIC_GRADIENT_DESCENT)
    .iterations(1)
    .updater(Updater.ADAM)
    .regularization(true)
    .l2(1e-5)
    .weightInit(WeightInit.XAVIER)
    .learningRate(2e-3)
    .activation(Activation.RELU)
    .list()
    .layer(0, new DenseLayer.Builder().name("encoder0").nIn(784).nOut(300).build())
    .layer(1, new DenseLayer.Builder().name("encoder1").nIn(300).nOut(30).build())
    .layer(2, new DenseLayer.Builder().name("decoder0").nIn(30).nOut(300).build())
    .layer(3, new OutputLayer.Builder().name("output").nIn(300).nOut(784)
        .lossFunction(LossFunctions.LossFunction.MSE).build())
    .pretrain(true).backprop(true)
    .build();

```

Fig. 5.5 Autoencoder detailed configuration.

5.4 Network Parameters and Training Process

Neural network needs some pre-configuration to get the relative best performance in accuracy, and this kind of pre-configurations is also called hyperparameters. For the selection of those hyperparameters, we did lots of experiments to try different setting and different result. As a result, rectified linear unit (ReLU) is set as activation function, and we perform 30 epochs training procedure for the consideration of prevent overfit when training. For intuitive impression we plot the training process of the output in every iteration, here we show different stages of training process in Fig. 5.6.

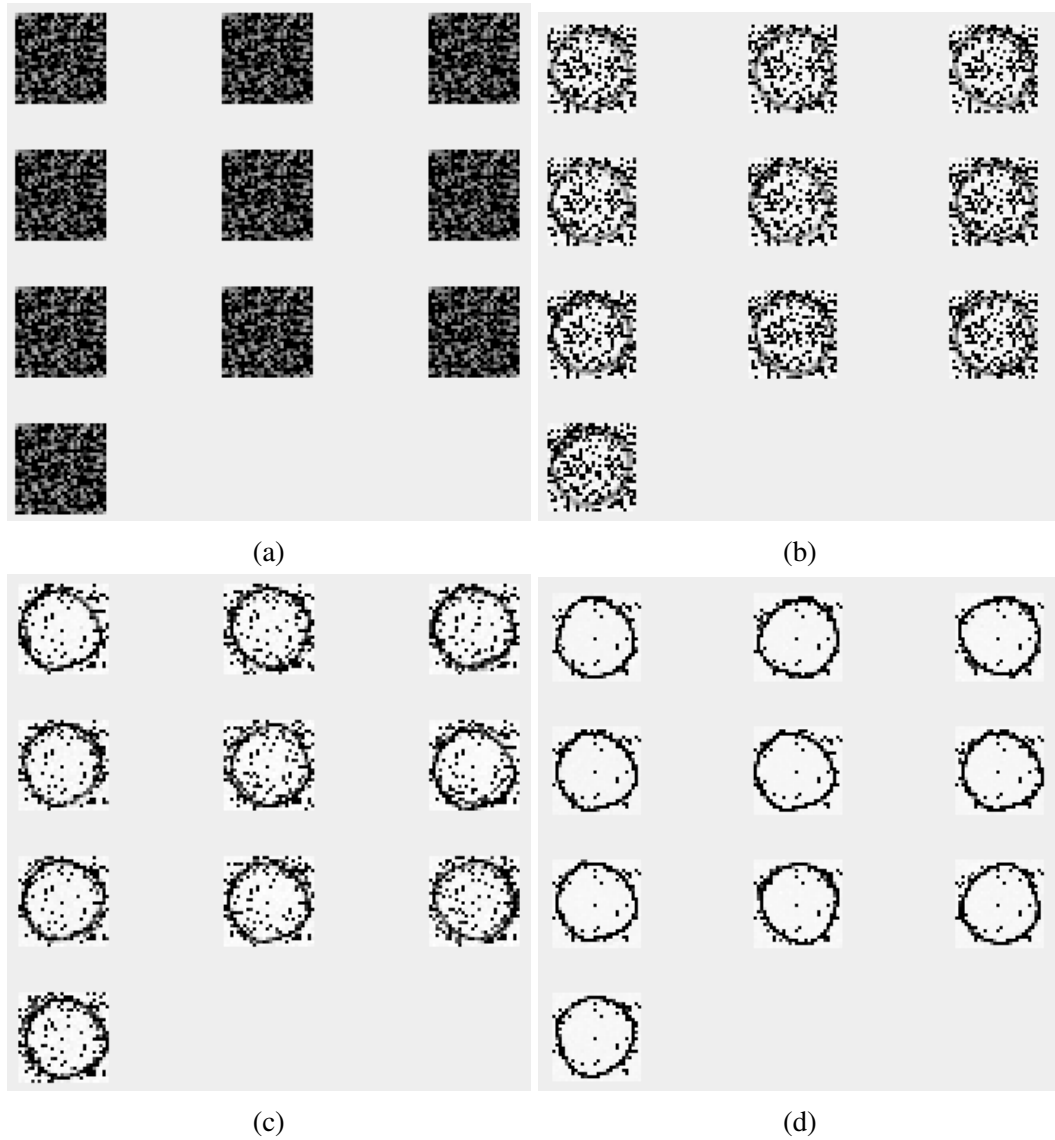


Fig. 5.6 The different training stages of 6000 iterations, the input gesture is gesture “circle”, but this figure only shows plane X-Y’s training. X-Z, Y-Z’s training looks different because there are three autoencoders to training them separately. (a) is in the initialization stage (iteration=1); (b) is the figure when iteration=500; (c) is the figure when iteration=2000; (d) is the final stage (iteration = 6000).

Fig. 5.6 shows the training process of different iterations in gesture “circle”. As the number of iterations increases, the output results of the autoencoder will be more and more fitting to the input, but the fitting is not equal to the zero error from input data, but with some errors in the reduction of the input. The establishment of this model provides a standard reductive output for specific inputs. As mentioned in the previous part, the output and

input can be compared, and the standard of difference is MSE score. The lower MSE score indicates that input and output are the more similar from shape while the higher the MSE score indicates the greater difference.







Class	Input	Output	Score
Positive			0.062412
Negative			0.154253
Negative			0.293413

Fig. 5.7 Different scores of different inputs

Fig. 5.7 gives the different scores in different tests come. It is remarkable that the input at Fig. 5.7 is test dataset not training dataset. The output is produced by a trained autoencoder (as Fig. 5.6 shows). From that figure, the more similar between the testing and training dataset, the higher reduction degree will be got, the output will be more like the training dataset. If the input of the test dataset does not match the input of the previous training, the output will become very chaotic.

5.5 Classification and Incremental Learning

When our neural network is trained, the boundary is automatically generated by our boundary strategy according to the record of training process. When the system is being used, if a new data is entered into our system, the encoding and decoding process will revisit, and then generate the MSE score according to the comparison between output and input. According to this score, the classifier will determine whether the score is higher than the boundary or below the boundary, making a positive or negative judgment.

Once the positive classification is made, this time of data will be regarded as a new batch of training data to re-train the network. It is an updating process which we called as incremental learning. In that process, the learning rate will be set as a lower value (0.0005) that initialization training process (0.002), and also the epoch times will be decreased a lot to 10 epochs. That makes it possible to make the network remember the new knowledge at the same time not forget the original knowledge.

This incremental learning process will help a lot in continue using of our system, because the more you use the system the more accurate your system will be. The experiment will be given on experiments parts.

5.6 Graphical User Interface

For the user-friendly usage and testing our approach, we also developed a graphical user interface for using which shown in Fig. 5.8.

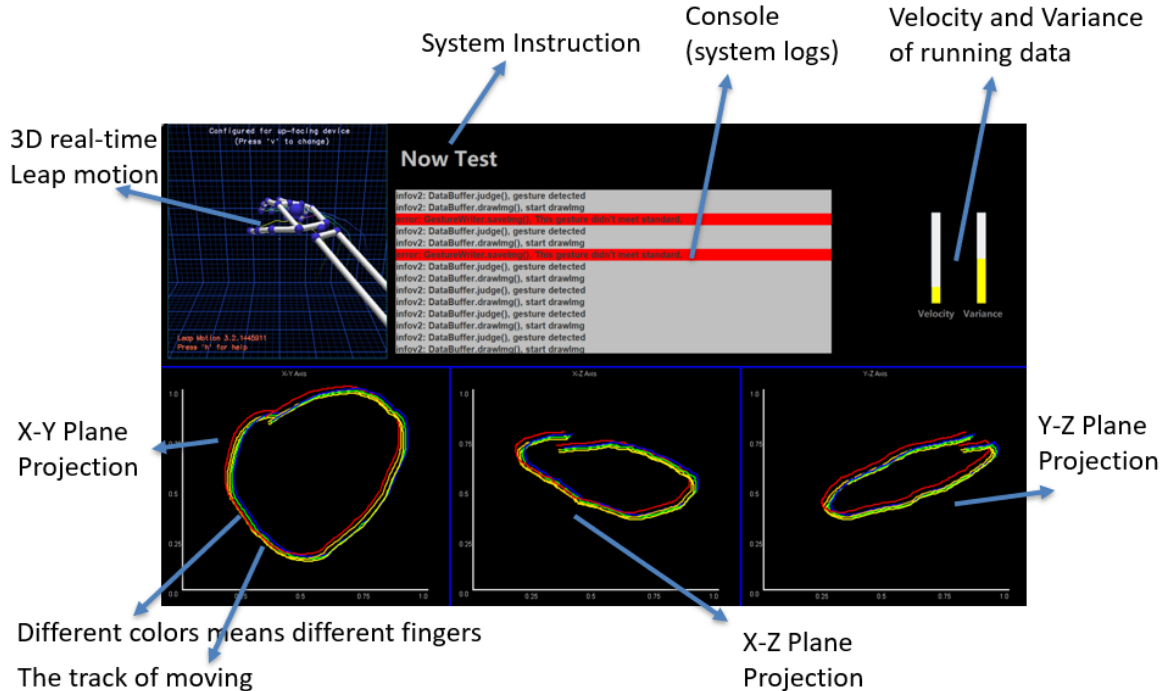


Fig. 5.8 Graphical user interface

The GUI is divided into two parts. On the top part, the left side is 3D real-time hand model; then is the console area which shows the system working information and instruction of system is on the top of console; the right side is velocity and variance bar of running data. On the bottom part, there are three panels which indicate three different planes of mapping data.

This design has mainly two purposes: for users and for developers. For users, it is very easy to look at your own gesture very directly and you could read the instruction as the running state. For developers, testing this system also needs to read the full information of current running data situation.

Chapter 6

Related Work

The related work will be introduced in this part. There will be two kinds of related work, the first is the work about gesture authentication and the second is about OCC problem and autoencoder.

6.1 Related Work about Gesture Authentication

Within our knowledge, there are a number of researches we could find, according to their different input devices, there are several types of gesture authentication systems.

Clark et al. [9] did some researches on engineering gesture-based authentication. They did the survey for authentication possibilities on security and possible methods. The conclusion of their research gave the bright prospects on gesture authentication.

Chong et al. [10] used mobile phone sensors to create templates by capturing different parameters of gestures. They had performed lots of gestures and then established different templates for different users. Also the experiments were done for verify the accuracy of their approach.

Kinect is also a kind of depth camera. Shukran et al. [12] proposed a Kinect-based gesture authentication system. They employed Baum-Welch algorithm to classify the different gestures. Liu et al. [20] also proposed a system using dynamic time warping algorithm and

using 3D accelerator, also a series of experiments had been done to evaluate their algorithm's performance.

Sae-Bae et al. [11] proposed an approach using multitouch gesture-based authentication using feature computation and distance function strategy. By using performance evaluation metrics, they got the score of EER 7.88%.

Aumi et al. [21] proposed a high-performance of gesture authentication, they used a distance-based template matching algorithm to make gesture authentication, the EER would be 0.029. They were also trying to make their system very easy to use.

Kamaishi et al. [22] used Leap Motion as input device and they gave lots of users cases about how gesture authentication could be used in realistic world. They used a series of comparison to match the gesture if this gesture is user's or not. Zhao et al. [23] also used Leap Motion using hidden Markov model as classification method to make multi-class classification, and they had got the result of FAR 1.65% and FRR 4.82%. Aman Chahar et al. [13] proposed a system also based leap motion and they used conditional mutual information maximization algorithm to select the optimal feature set. Match-score fusion was performed to reconcile information from multiple classifiers. They had achieved 81.17% Recall on FAR 1%. They had also compared different methods and evaluated their approach.

In summary, there are several researches about gesture authentication. The device, algorithm, user cases have many differences but also share some commons. The commons mainly reflect in the accuracy of the algorithm, because authentication itself is a very strict thing which needs a very high accuracy, making the authentication very security.

6.2 Related Work about One-Class Classification and Autoencoder

One-class classification is a special situation in classification technologies. For making the result better we've done lots of studies on this field. Here are some researches on OCC and autoencoder.

The historical research on autoencoder to do classification was proposed by Japkowicz et al. [18]. There were two keys on this paper, the first the autoencoder was mentioned to do classification, the second was that they gave a threshold determination method for classification. Also they did some case studies about the comparison between different methods.

Khan et al. [6] gave the survey on recent trends in one-class classification. They sorted the different methods in one-class SVM and other methods as non-OSVM, also they had given the comparison on different methods. In one-class SVM, Schölkopf B et al. [24] proposed this algorithm and then this algorithm had been known by others. Y Chen et al. [25] proposed that using one-class SVM in image interval. In non-OSVM, S Ramaswamy et al. [26] proposed an algorithm that mining outliers from large dataset. This algorithm ranked the different point based on the Euclidean distance from different points.

Manevitz et al. [7] gave the research that using autoencoder on document classification. They presented 6 methods on threshold strategies and evaluated the performance of them. Their approach was very inspirational, they made a lot of evaluations, and compared their method to other one-class classification methods, the conclusion was that their method had better performance than other methods.

Autoencoder used to solve one-class classification is the thing occurred recently, and the approaches used in human computer interaction research field are still limited. Compared to their approaches, our research mainly wants to use this type of neural network as a dimension expression method, reducing from a very high-dimensional data to a just one-dimensional score interval, then according to this interval, we could classify the positive class and negative class.

Chapter 7

Experiment and Result

In order to prove the performance on authentication, the experiments aim to verify the accuracy of user's gesture and the accuracy when false gestures come. To realize that experiment thinking, we will perform an experiment and the result will also be shown.

The experiment environment is based on our system we have already developed as shown in Fig. 6.1. We have built a system to do gesture authentication, which is described on the Chapter 4 and Chapter 5.

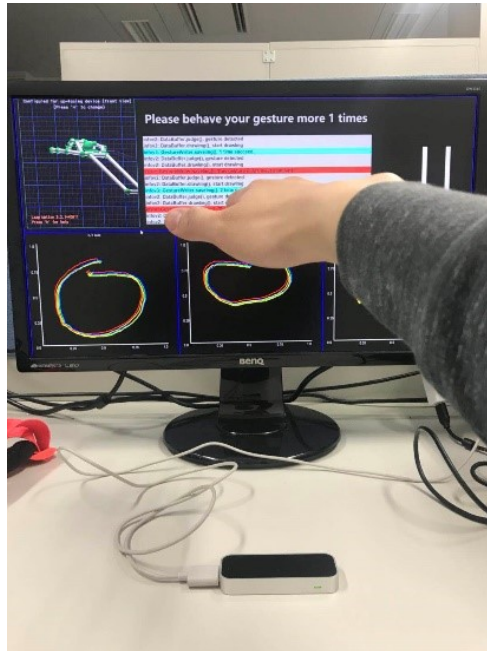


Fig. 7.1 The gesture authentication system we developed for using and test.

7.1 Experiment

Before doing the experiment, we have searched lots of public database for the dataset we needed, but within our knowledge that we cannot find such a dataset of the requirement we need, so we would construct the experiment dataset by ourselves. We have designed an experiment to evaluate the performance of our approach. We asked 6 users of 22~25 years old who had no experience of this kind of system but they were taught about how to use our system:

1. At registering stage, 2 users will be asked to behave simple gesture, 2 users will be asked to behave normal complexity gesture and 2 users will be asked to behave complicated gesture. Fig. 7.2 shows the sample of simple, normal and complicated gesture; (Every user will behave 3 times)
2. After that we will ask them to authenticate their own gesture 20 times;
3. Then each user's gesture will be tested by other 5 users, each user will test 4 times, totally $4*5=20$ times for one user. This is an attack experiment.

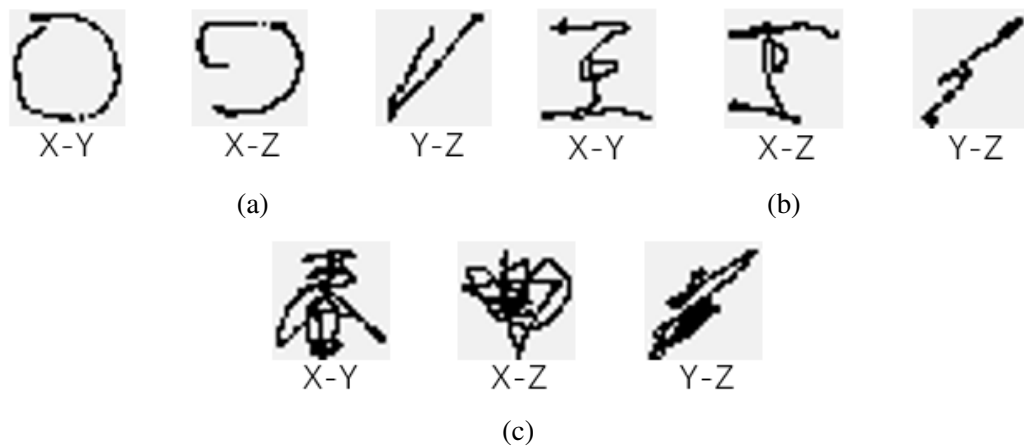


Fig. 7.2 The sample of gestures. (a) is the simple gesture, (b) is the normal gesture and (c) is the complicated gesture.

Note that in every stage, their original data will be saved for further analysis. From the experiment, we will analyze the result by:

1. FAR, FRR score of every user's gesture as basic accuracy analysis;
2. Comparison of simple, normal and complicated gesture;
3. Comparison of different stage of incremental learning;
4. Comparison of using, using partially and not using data augmentation;
5. Comparison of our approach and previous works;

7.2 Result

For choosing the evaluation standard, we use false acceptance rate (FAR) and false rejection rate (FRR) as performance index, and for intuitive and detailed view, we use receiver operating characteristic (ROC) curve for further explanation.

If the model classifies the gesture as the positive gesture and actually this gesture itself is also positive gesture, this time of classification will be a true positive (TP) classification. Respectively, if the model classifies the gesture as negative and the gesture actually is negative, it will be regarded as true negative (TN) classification; False positive (FP) is the false classification that the predicting the negative gesture to positive and false negative (FN) is predicting the positive gesture to negative. FAR and FRR can be defined as follows: $FAR = FP/(FP+TN)$ and $FRR = FN/(TP+FN)$.

Receiver operating characteristic (ROC) curve is a kind of curve which is a comprehensive indicator reflecting continuous variables of sensitivity and specificity. The curve usually is used to evaluate the performance of a classifier, where the horizontal axis is false positive rate (FPR) score and vertical axis is true positive rate (TPR).

7.2.1 False Rate Analysis

In this section we will analyze the basic accuracy of these 6 users.

We give the result of our experiments is shown in Table 7.1, which is the result after training (which means just register already, 3 times gesture behaving) and after 20 times pass (which means the model has been incrementally retrained 20 times).

Table 7.1 FAR and FRR score of different gestures after training

User ID	after just training		after 20 times pass	
	FAR	FRR	FAR	FRR
1	0.90%	4.70%	0.54%	0.74%
2	1.24%	4.09%	0.69%	1.08%
3	1.82%	3.46%	0.67%	1.18%
4	1.53%	3.66%	0.62%	1.02%
5	2.31%	2.73%	0.82%	1.06%
6	2.26%	2.82%	0.79%	1.22%
Avg.	1.68%	3.58%	0.69%	1.05%

7.2.2 Comparison of Simple, Normal and Complicated Gesture

In this section, we will show the performance on the different complexities of gesture: simple, normal and complicated. We used the original data of 6 users' gesture, and by manually adjusting the threshold value we generated ROC curve shown at Fig. 7.3.

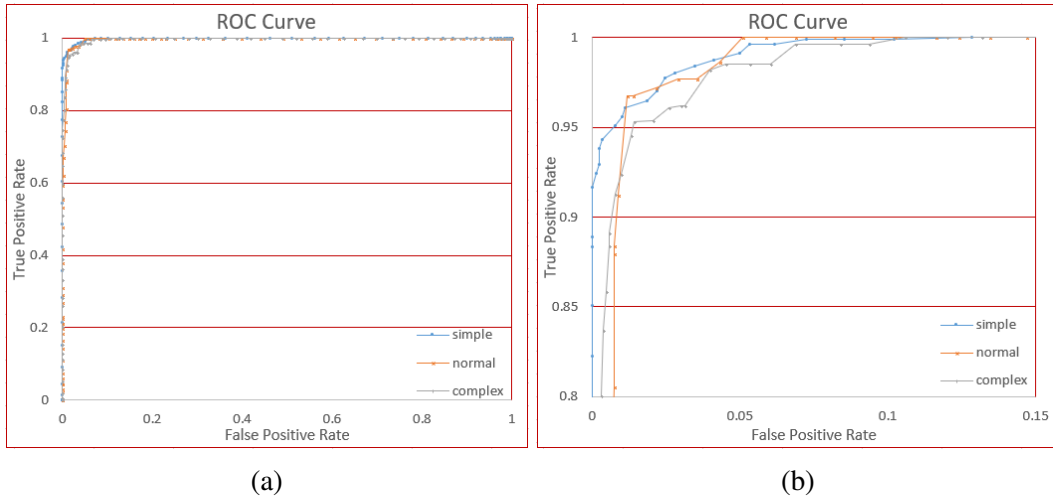


Fig. 7.3 The ROC curve of simple, normal and complicated gesture. (a) is the global view and (b) is the local view for clarity.

From the Fig. 7.3, we could see that the curve fits almost to the upper left corner. In the ROC curve, the closer the curve is from the upper left corner, the better the classifier is.

For three different complex gestures, we can see the TPR (=1-FRR) value is higher than 95% where FPR (=FAR) is near 1% from the curve. We can see no matter the high and low complexity of the gestures, our approach has a very good performance. Also, as a result, the FPR performance of simple is very good, on normal and complex gesture, the performance is also acceptable.

What beyond our expectation is that the simple gesture has the better result than more complex gesture, we did some survey and guessed the reason may be that a small change will produce a big difference.

7.2.3 Analysis on Data Augmentation

In this section, we analyzed our approach's performance on data augmentation. We thought it is very important to evaluate if data augmentation will affect the performance or not.

We invited the user performing the "circle" gesture and asked him to help us more. The original "circle" gesture used data augmentation by behaving 3 times, he was asked to do more: 15 times and using just shift operation; 60 times and without any data augmentation operation.

Table 7.2 shows the FAR and FRR of these three groups. From that we can see, using or not using data augmentation has only a little influence of the FAR and FRR. That is because our boundary strategy makes the threshold according to the different situation of every gesture.

Table 7.2 FAR and FRR score of using or not using data augmentation (DA).

Type	FAR	FRR
DA & 3 times	1.24%	4.09%
Shift Only & 15 times	1.67%	4.21%
None & 60times	0.97%	4.80%

To deeply study the influence of data augmentation, we also draw ROC curve as shown in Fig. 7.4.

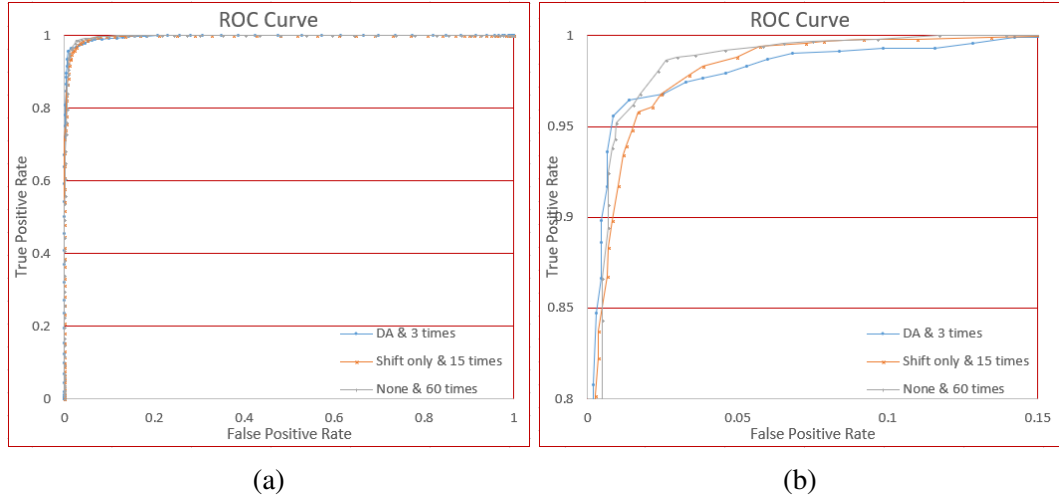


Fig. 7.4 The ROC curve of using or not using data augmentation. (a) is the global view and (b) is the local view for clarity.

From Fig. 7.4 we can see the difference of using and not using data augmentation. The curve of these 3 curves is different. The third curve (not using data augmentation) has a little better performance, followed by the second curve (only using shift operation), and first curve (using full data augmentation) is still not far away from other curves. This figure proves the effect of data augmentation on accuracy is negligible.

7.2.4 Analysis on Incremental Learning

This section we showed the analysis on incremental learning. Incremental learning is also based on user-friendly consideration, but unlike data augmentation technology, incremental learning will increase system accuracy by using and using to achieve high reliability, we will compare the model after training with models that use incremental learning for a period of time.

Three groups will also be made for "circle" gesture: first group is just after training, then we will use the model by 5 times' passes (which means 10 times' incremental learning will be done) as the second group and the third group is after 20 times' passes.

Table 7.3 shows the FAR and FRR of different incremental learning time. To our surprise, the result after incremental learning is much better than before incremental learning. The result actually has already meet the standard of strict authentication scenario like password.

Table 7.3 FAR and FRR score of different incremental learning times (ILT).

Type	FAR	FRR
ILT=0	1.24%	4.09%
ILT=5	1.10%	1.45%
ILT=20	0.69%	1.08%

Also, for intuitive impression, we also draw the ROC curve for analysis. The curve is shown in Fig. 7.5.

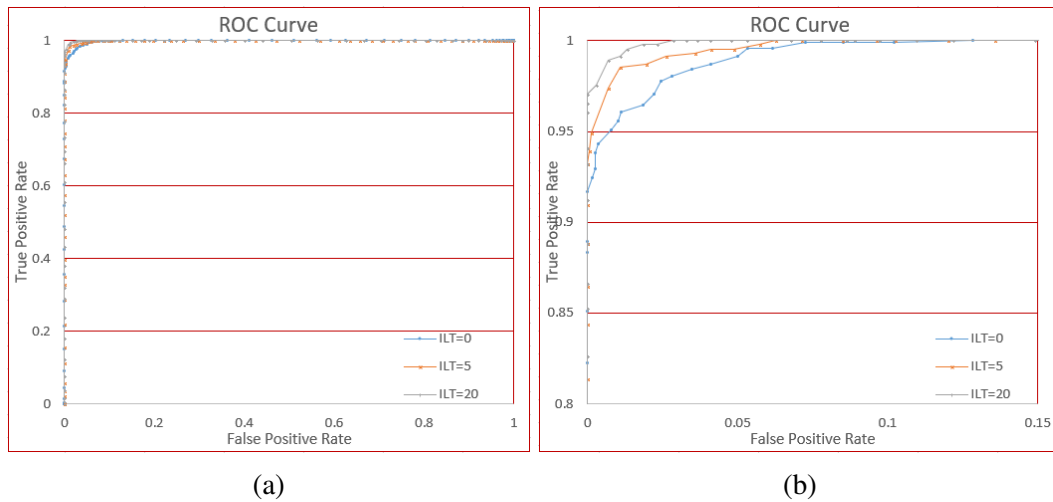


Fig. 7.5 The ROC curve of different incremental learning times (ILT). (a) is the global view and (b) is the local view for clarity.

From Fig. 7.5 we can see the more using the system, the higher the accuracy will be. This is also the meaning of incremental learning. 20 times of using this system will almost remember your own characteristics.

7.2.5 Comparison of Previous Works

Within our knowledge, researchers have contributed a lot in this field. We summary some of their works and make the comparison to our work in Table 7.4.

Table 7.4 Comparison of previous works and our approach.

Reference	Sensor	Algorithm	FAR	FRR	EER	Accuracy
[23]	Leap Motion	HMM ¹	1.65%	4.82%	-	95.21%
[21]	Intel Senze3D	DTW ²	-	-	2.9%	94.3%
[20]	3D accelerometer	DTW	-	-	-	98.4%
[27]	Tablet	SVM ³	1.2%	2.6%	-	over 98%
[28]	Tablet	ESM ⁴	-	-	-	98.90%
[13]	Leap Motion	CMIM ⁵	1%	18.83%	-	-
[29]	Smartphone	DTW	0.27%	4.65%	1.86%	-
Ours	Leap Motion	AE ⁶	1.68%	3.58%	-	-

¹ Hidden Markov Model² Dynamic Time Warping³ Support Vector Machine⁴ Experience Sampling Method⁵ Conditional Mutual Information Maximization⁶ Autoencoder

From the table we could see different study uses different evaluation standard, it is very difficult to use the same standard to measure these studies. Roughly just seeing the performance, our approach that incremental learning time (ILT) equals to 0 (means just registered) just has a little better performance among them; But after 20 times of incremental learning, our approach performs a very low false rate among the studies.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

In this thesis, we proposed a system which used gesture movement as authentication method and employed many improvements on minimizing the user's effort.

Biometrics authentication is very popular in recent authentication. Not only the accuracy needs to be satisfied but also the user-friendly consideration should be satisfied. But previous researches have the problem of short of these consideration. Our target system should satisfy there 3 requirements: could be used by just single user's customized gesture; could be used without too many times of behaving gesture but it will still have high accuracy; could be used to continue learning user's gesture while user is using.

We use Leap Motion as the input device which is a infrared based depth camera. By continuing reading the data from Leap Motion, we first filtered these data with Median Filter and Gaussian Filter to remove some noise caused by device. Then we normalized the input gesture into a same standard for further process. For the consideration of using 3D gesture, we map the 3D gesture information into 3 plane: X-Y, X-Z and y-Z. Then for reducing the time of user behaving, we used data augmentation. For doing one-class classification, we used autoencoder to generate the output image from the certain input image, and to compare their difference by mean squared error function. Then for classifying the positive and negative class, we proposed a boundary making strategy for classification. Once the

system is under using, if a user passes his system, the new knowledge will be retrained as incremental learning. For proving the performance of our approach, we performed an experiment to evaluate the false rate of our system. After training our system has FAR 1.24% and FRR 4.09%, after 20 times of using, our system has FAR 0.69% and FRR 1.08%. It is enough to be and behavior biometrics authentication method [30].

8.2 Future Work

Since our system is out of the user-friendly consideration, there are lots of application scenarios. Compared to the traditional text password, we are thinking if there is one possibility that will replace text password with more user-friendly biometrics methods. In static physiological characteristic field, fingerprint is a very successful example to be an authentication method. As talking about the behavior characteristic, it still cannot ensure 100% accuracy. In some secret applications such as bank account application it still needs more exploration. But we believe that the potential of behavior characteristic will grow more and more and finally it will become a very popular and reliable authentication method.

References

- [1] Anil K Jain, Ruud Bolle, and Sharath Pankanti. *Biometrics: personal identification in networked society*, volume 479. Springer Science & Business Media, 2006.
- [2] Davide Maltoni, Dario Maio, Anil K Jain, and Salil Prabhakar. *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.
- [3] Jeff Friedman. Muscle memory: performing embodied knowledge. In *Text and Image*, pages 156–180. Routledge, 2017.
- [4] Simon Fong, Yan Zhuang, and Iztok Fister. A biometric authentication model using hand gesture images. *Biomedical engineering online*, 12(1):111, 2013.
- [5] Sushmita Mitra and Tinku Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(3):311–324, 2007.
- [6] Shehroz S Khan and Michael G Madden. A survey of recent trends in one class classification. In *Irish Conference on Artificial Intelligence and Cognitive Science*, pages 188–197. Springer, 2009.
- [7] Larry Manevitz and Malik Yousef. One-class document classification via neural networks. *Neurocomputing*, 70(7-9):1466–1481, 2007.
- [8] Shikha Agrawal and Jitendra Agrawal. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, 60:708–713, 2015.
- [9] Gradeigh D Clark and Janne Lindqvist. Engineering gesture-based authentication systems. *IEEE Pervasive Computing*, 14(1):18–25, 2015.
- [10] Ming Ki Chong and Gary Marsden. Exploring the use of discrete gestures for authentication. In *IFIP Conference on Human-Computer Interaction*, pages 205–213. Springer, 2009.
- [11] Napa Sae-Bae, Nasir Memon, Katherine Isbister, and Kowsar Ahmed. Multitouch gesture-based authentication. *IEEE transactions on information forensics and security*, 9(4):568–582, 2014.
- [12] Mohd Afizi Mohd Shukran and Mohd Suhaili Bin Ariffin. Kinect-based gesture password recognition. *Australian Journal of Basic and Applied Sciences*, 6(8):492–499, 2012.

- [13] Aman Chahar, Shivangi Yadav, Ishan Nigam, Richa Singh, and Mayank Vatsa. A leap password based verification system. In *Biometrics Theory, Applications and Systems (BTAS), 2015 IEEE 7th International Conference on*, pages 1–6. IEEE, 2015.
- [14] David A Van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, 2001.
- [15] David A Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *International journal of computer vision*, 77(1-3):125–141, 2008.
- [16] J Sola and Joaquin Sevilla. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on nuclear science*, 44(3):1464–1468, 1997.
- [17] David A Van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, 2001.
- [18] Nathalie Japkowicz, Catherine Myers, Mark Gluck, et al. A novelty detection approach to classification. In *IJCAI*, volume 1, pages 518–523, 1995.
- [19] Eclipse Deeplearning4j Development Team. Deeplearning4j: Open-source distributed deep learning for the jvm, apache software foundation license 2.0. <http://deeplearning4j.org>.
- [20] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uwave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, 2009.
- [21] Md Tanvir Islam Aumi and Sven Kratz. Airauth: evaluating in-air hand gestures for authentication. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*, pages 309–318. ACM, 2014.
- [22] Satoshi Kamaishi and Ryuya Uda. Biometric authentication by handwriting using leap motion. In *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, page 36. ACM, 2016.
- [23] Jinghao Zhao and Jiro Tanaka. Hand gesture authentication using depth camera. In *Future of Information and Communication Conference*, 2018.
- [24] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588, 2000.
- [25] Yunqiang Chen, Xiang Sean Zhou, and Thomas S Huang. One-class svm for learning in image retrieval. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 34–37. IEEE, 2001.
- [26] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *ACM Sigmod Record*, volume 29, pages 427–438. ACM, 2000.

- [27] Yimin Chen, Jingchao Sun, Rui Zhang, and Yanchao Zhang. Your song your way: Rhythm-based two-factor authentication for multi-touch mobile devices. In *Computer Communications (INFOCOM), 2015 IEEE Conference on*, pages 2686–2694. IEEE, 2015.
- [28] Yulong Yang, Gradeigh D Clark, Janne Lindqvist, and Antti Oulasvirta. Free-form gesture authentication in the wild. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3722–3735. ACM, 2016.
- [29] Ziwen Sun, Yao Wang, Gang Qu, and Zhiping Zhou. A 3-d hand gesture signature based biometric authentication system for smartphones. *Security and Communication Networks*, 9(11):1359–1373, 2016.
- [30] Weizhi Meng, Duncan S Wong, Steven Furnell, and Jianying Zhou. Surveying the development of biometric user authentication on mobile phones. *IEEE Communications Surveys & Tutorials*, 17(3):1268–1293, 2015.