A case study of the Robustness and the Usability of CAPTCHA

Tianyi Hu

University of Tampere
School of Information Sciences
Software Development M.Sc. thesis
Supervisor: Zheying Zhang
May 2016

University of Tampere
School of Information Sciences
Software Development
Tianyi Hu: A case study of the Robustness and the Usability of CAPTCHA
M.Sc. thesis, 61 pages, 8 index and appendix pages
May 2016

The websites and network application experienced explosive growth in the past two decades. As the evolution of smartphones and mobile communication network have evolved, smart phone's user experience has been improved to a high level, and more and more people prefer to use smartphones. However, the development of techniques will not only increase the users' experience but also bring threats of cracking. The development of techniques brought the potential threats to websites' security. As a result, CAPTCHA, Completely Automated Public Turing test to tell Computers and Humans Apart, forms one of the methods to impede spamming attacks.

As CAPTCHA's definition indicates, CAPTCHA should be recognized by humans easily while shouldn't be recognized computers. These two attributes of CAPTCHA can be considered as usability and robustness. Some CAPTCHA is difficult to be recognized by computers, but humans may also find difficult to recognize it. Therefore, the purpose of the thesis is to find out the balance between usability and robustness of CAPTCHA. Therefore, the related researches about the usability and the robustness of CAPTCHA will be reviewed, and the process of automatic CAPTCHA recognition will be Figured out and implemented by the author. The implementation will be based on the existed algorithms and a case study.

The findings are the factors for improving CAPTCHA's robustness. They are from the each step of a specific process of automatic CAPTCHA recognition. Then the factors will be compared with the issues which are from the related usability research. The discussion will derive some possible ways, such as adding confusing characters and increasing data's diversity to improve robustness while keeping the usability according to the derived factors.

Key words and terms: CAPTCHA, usability, robustness, machine learning.

Contents

1. Introduction

1.1. Motivation

Web sites and network applications have experienced explosive growth in the past two decades [29]. As the evolution of smartphones (e.g. Samsung and iPhone) and the mobile communication technology, smart phone's user experience has been improved to a high level and more and more people prefer to use smartphones. However, the development of technology will not just increase the users' experience but also the convenience of cracking. The development of technology, especially the development of the network, has brought potential threats to websites' safety and users' privacy since the cracking program can attack more times at the same time than before.

One method to avoid web service invasion and spamming is CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart [8]), and it became more and more popular. CAPTCHA is a tool to generate a test which is readable for people but cannot be recognized by computers. CAPTCHA often appears when users want to get information or submit information to the server (backend) of websites. As a result, CAPTCHA is an additional authorization process of users who can easily get through the process and robots which will cost much to go through the process.

For example, when a user creates a new account, CAPTCHA will usually appear before submitting the application form. When creating a new account, the data interaction will be made between the web pages (frontend) and the database (backend). If there is no CAPTCHA, it is possible to submit forms constantly via an auto-submitting program. The database, as well as the server, will process a huge number of the data as if a million people send their new account creation request at the same time. The processing data is so large that it will very possibly paralyse the server and force database to refuse all the requests to protect itself. As a result, normal users will hardly submit forms to the websites and even cannot refresh the site. In this thesis, CAPTCHA refers to the CAPTCHA schemes or a specific CAPTCHA method. Furthermore, "attacker" refers to the cracking robots as well as the robots' designer.

Many researchers have paid much attention to character recognition in the CAPTCHA for the past 15 years [29, 40, 45, 46]. Currently, it still plays an essential role in protecting web services from abuse, even if the intelligent character recognition has developed in recent years [29]. However, some web services still choose the basic CAPTCHA generation algorithm as it costs fewer resources on the server and is easy to implement. It indicates that the owners of the websites have poor awareness of their internet safety.

CAPTCHA can reduce the attacks which can sometimes steal users’ data from the websites’ server. It can also impede a brute-force attack which systematically checking possible passwords until it finds the the correct one. Furthermore, personal privacy is important, and privacy needs to be secured, especially in the websites that handle sensitive personal data. If brute-force attacks attack the websites, it may result in significant loss.

A CAPTCHA generation tool is an automatic program which can distinguish whether the guests are human beings or a computer. CAPTCHA can be added as an additional authorization process to the websites, and it can prevent data from being stolen, websites from being attacked by DDoS (Distributed Denial of Service) attacks, spamming attacks and other unauthorized invasions. If an application can form questions which cannot be answered by a computer, then the guests who can respond to the questions can be regarded as human.

CAPTCHA is a common method for protecting the websites' server from spamming attacks and other invasions. There are many popular websites such as Facebook[1] and Twitter[2] requiring guests to input the identifying codes when they log in or submit comments. However, sometimes the identifying codes are so simple that their contents of the image can be recognized automatically, even if it is the text-based CAPTCHA. It informs guests of the question to which the server has the answer and a static picture with some noises (sometimes even no noises) in it. The guests who give the correct answers can be regarded as human. Nevertheless, since computer handles the questions and judgments, they may be cracked easily by using automatic recognition methods such as optical character recognition (OCR).

1.2. Research Questions

There are two main requirements for CAPTCHA---easy for humans to recognize; difficult/impossible for computers to recognize. As for the humans part, it can be regarded as usability since the humans part is oriented to human beings. On the other hand, the computers part is about the robustness. Thus, the aim of this thesis is to improve CAPTCHA design and generation. Moreover, this research seeks to answer the following questions:

- By existing CAPTCHA recognition approach, what are the factors impacting on CAPTCHA’s robustness?

[1] http://www.facebook.com

[2] http://twitter.com

- How to make a trade-off between the usability and the robustness of a CAPTCHA generation algorithm?

At first, CAPTCHA could be considered as a product since it can fulfil customers' requirements and benefits merchants with its added value. The value comes from protecting sensitive personal data as well as protecting websites from hack attacks so that customers can feel safe about their personal data and merchants can enhance the profit with the added value from CAPTCHA.

Besides value and actual functions, products have many attributes that may change the value of the product, such as usability, reliability, robustness and other non-functional attributes. These attributes can obviously impact the user experience. Moreover, these attributes can dramatically improve the product's value even if the functions are mediocre and replaceable. For example, the usability and the robustness of CAPTCHA are essential when generating CAPTCHAs. However, usability and robustness may have differences in their definition in details.

In this thesis, the robustness of CAPTCHA refers to the strength to get rid of the cracking behaviours or being recognized by computers. Moreover, it can also be considered as "security." In addition, the usability of CAPTCHA means the ease of recognizing CAPTCHA by human beings.

Then, as for the first question, it is easy to make the images difficult to be recognized by adding many complicated hot pixels, or combining the characters together or using other simple ways to make the pictures "ugly." Hot pixels are the isolated noise points in a picture, they can create much useless information to computer's recognition and can make the computer's recognition harder via distorting the characters. Unfortunately, that is insufficient as CAPTCHA requires not only that generated questions should be impossible to be recognized by a computer, but also that humans should acknowledge them correctly and effectively. This is also a reason why many websites choose a simple method of CAPTCHA.

The balance between the difficulties of automatic recognition and artificial recognition is the largest challenge for CAPTCHA. When people register in, for example, Microsoft[3]website, in 2011, the identifying codes are tough for humans to recognize [25]. Sometimes the pages need to be refreshed to find an easier one. However on some e-

[3] http://www.microsoft.com

commerce websites, for example, PayPal[4], in 2011, its identifying code images are so simple that the probability of cracking the code has almost reached 88% [25].

As a result, it is a challenge for the modern websites to make CAPTCHA easy to be recognized by humans. Meanwhile, it should cost much when using a computer to recognize CAPTCHA. Therefore, the question “How to make a trade-off between the usability and the robustness of a CAPTCHA?” forms another focus of this thesis.

1.3. Research Methods

Robustness and usability are both unintuitive, therefore, to answer the first research question, the concepts of usability and robustness will be discussed according to the related literature at first. As the second research question is focusing on the recognition process and algorithm which is more concerned with robustness, the usability part will be mostly explained via the literature review. To test the robustness of CAPTCHA and to answer the second research question, some related literature will be reviewed. Furthermore, a character recognition algorithm will be investigated. In this research, the machine learning algorithms will be taken into account. The CAPTCHA on captchas.net will be chosen as the test CAPTCHA [1] for the case study. It is easy to be processed by the recognition program because it uses the identifying code with static pictures. More analysis and explanation of this CAPTCHA will be given in Chapter 4.

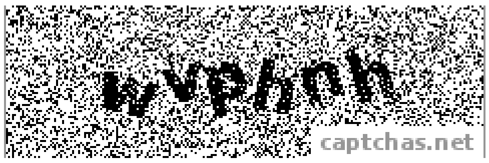


Figure. 1 CAPTCHA on captchas.net

There are three classifier algorithms to be introduced in Chapter 3 to improve the recognition accuracy. Several classifier ensemble methods have been proposed in many sources [2]. One of these sources emphasizes the complementary property of the features in classification combination by training the classifiers with features of obvious difference principal components [3]. The Validation methods and tools for recognition accuracy will also be introduced in Chapter 3.

Artificial recognition, being related to usability of CAPTCHA, is complicated due to many factors. Usability in real products was difficult to be academically analysed because analysts who are responsible for system requirements are not experts in this field --- human behaviours or software psychology [6]. As a result, this research will review the

[4] https://www.paypal.com

related articles and thesis to find out the usability factors of CAPTCHA. Then the first research question will be answered by the literature.

This thesis will put more emphasis on the algorithm. Thus, little data can be collected from the related publications. However, the robustness, which is the most relevant standard of a security system, of the generating image algorithms can be examined by a self-made recognition program. This thesis uses machine learning method in the recognition program. The learning data, CAPTCHA schemes, will be collected. The more pictures are collected; the higher accuracy will be. Therefore, the data for testing robustness will be quantitative.

In addition to the usability of CAPTCHA, the factors of user satisfaction for CAPTCHA can also be collected from the related literature. This can help to increase awareness of the balance between the difficulties of recognition by machines and human beings.

After gathering these data, finishing the automatic recognition, and comparing the elements that can influence humans' cognition of CAPTCHA and the cost of recognizing CAPTCHA automatically, the results proposes some factors for helping generate CAPTCHA will. Usability related research will be reviewed and discussed in the following sections.

The rest of this thesis is organised as follows. Chapter 2 presents the literature review on the basic contents of CAPTCHA, usability and robustness issues. Chapter 3 indicates the foundation of CAPTCHA recognition, the techniques to be utilized will be introduced, including digital image processing and machine learning. Chapter 4 demonstrates methodology with case studying and explains the process of recognizing CAPTCHA and test the robustness of CAPTCHA. Chapter 5 analyses the results of robustness test and usability research review, then has a discussion to find out the issues that can improve the robustness while keeping the usability. The limitation of the results will also be discussed here. Chapter 6 concludes the thesis.

2. CAPTCHA and its Usability and Robustness

Web security belongs to the concept of network and application security. It is classified as server-level security and user-level security [42]. Server-level security refers to the security of computers on which servers are running. It is important to secure Internet servers because they host all essential services being accessed from different parts of the world. User-level security is based on the browser security [42]. The security of network and privacy belongs to the server-level security.

The security of the network is facing serious threats. In considering of the principles of networks, attack methods could be created --- denial of service (DoS). DoS attacks will mainly attack web servers, application servers and communication links [42], therefore, all the users including the authorized users cannot use the network services. For example, an unauthorized user uploads quantity of useless data to a file transfer protocol (FTP) address. Then an unnecessary load of disk space and network traffic generation will be unusable services. According to the principles of networks, DoS includes activities: Disrupting the network traffic; disrupting the network connections; denying the server services to a client. DoS consists of two kinds of attacks, i.e. flood attacks and software attacks [42].

Flood attacks mainly focus on the network devices. For instance, routers and network information centres (NIC), whose capacity to process packets is limited, are the common targets of flood attacks. Therefore, network devices can be attacked utilizing the flaws of protocols. For example, SYN (synchronous) flood attacks are utilizing bugs of TCP (Transmission Control Protocol); Smurf flood attacks are based on ICMP’s (Internet Control Message Protocol) flaws; Fraggle attacks utilize UDP (User Datagram Protocol) as their basis. As these methods are attacking lower level of the network, CAPTCHA cannot deal with them [42].

However, as technology develops, the website/network application servers have better processing speed and larger internal memory. Therefore, it is easier to attack network via utility of the known software weaknesses. This method is called software attacks. DDoS is one mean of software attacks. It is the upgrade form of DoS and controls more botted machines to attack target via DoS at the same time. DDoS increases a load of the target server to geometric multiple times.

Besides the DoS, the spamming attack can not only realise DoS but also annoy common users via breaking functionality of the websites/applications. Spamming attack means the attacker sends hundreds or thousands of trash information to a website/application in a short time (e.g. five thousand messages in one minute). Some websites such as the BBS (Bulletin Board System) offer a commenting feature which allows users to write feedbacks and comments. It means anyone including robots can access to the website and submit content to the server. If the attacker uses robots to submit trash information automatically, the website will be a mess. Attributing to the garbage information includes commercial advertisements, meaningless messages, hyperlinks which lead to Trojan/virus websites. The three kinds of trash information will all annoy common users via disturbing them finding the information they want to see. Except for the meaningless messages, the other two kinds of trash information can also be submitted by human beings. Therefore, meaningless messages are the main method of spamming attack. Furthermore, meaningless messages will create a similar effect of DoS. Meaningless messages are easy to make and can be submitted by robots quickly as their small sizes. The meaningless messages can fill up the website so fast that common users may be not able to find the original information after refreshing the web page. Thus, it is also a kind of DoS as common users cannot use the services after being attacked by spamming attacks.

As a result, CAPTCHA, which can distinguish computers and human beings, is needed for impeding spamming attacks. If there is CAPTCHA when an unauthorized user tries to submit information, the robots' access will be denied by CAPTCHA. CAPTCHA can also avoid the misoperation, such as clicking "submit" button more than once, of the unauthorized people.

CAPTCHA is also a common method for protecting the websites' server from DDoS attacks. There are some popular websites requiring guests to input the identifying codes when they log in or submit comments. Besides CAPTCHA, some other methods could also be applied to avoid DDoS attacks. For instance, put the internet protocol addresses from where too many access information in a quite short time are (e.g. one hundred accesses in just three seconds) into the blacklist. Then the access from the internet protocol addresses of blacklist will be denied directly.

From the perspective of the privacy security, CAPTCHA can do well in preventing brute-force attacks. Brute force attack, which is also called exhausted key search, is a kind of password cracking. In the database, users' account name and passwords are encrypted by some algorithms, and the hash is one of the algorithms. In a brute force attack, the intruder first obtains a list of used passwords from bona fide sources; then the passwords will be hashed via encryption schemes supported by the system being attacked. The resultant encrypted hash values are then compared with the hash passwords stored in the database. The comparison is successful whenever the hash value from the dictionary matches the hash value in the database [42].

CAPTCHA can create an additional authorization process. Thus it requires brute-force attack to add CAPTCHA recognition function to the attack process. However, even if the CAPTCHA recognition function is added to the attacking process, the brute-force attack must take more time to get the passwords. If the accuracy of the recognition cannot keep a high level (over 90%), the number of attempts will increase dramatically. For instance, if the accuracy of CAPTCHA recognition is 80% and n is the original number of attempts, the expected value of number of attempts will be 1.25n (10/8 * n). If the accuracy of CAPTCHA recognition is 50%, the expected value of number of attempts will be 2n, which means the cracking time is doubled.

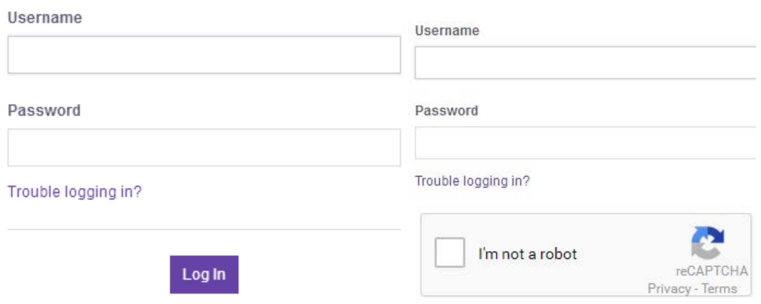


Figure. 2 login page on Twitch

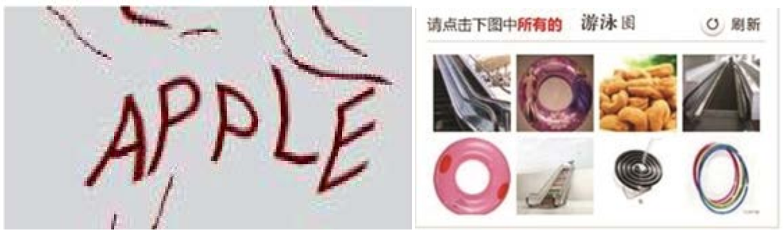
In Figure 2, it is the login page of Twitch[5]; it is a game live interactive platform. To be concreted, it is a platform that game players can broadcast his game play while the players are playing and other users can watch them play and give comments. Therefore, the server of Twitch should handle a large quantity of streaming data. The left picture of Figure 2 is the standard login page. The right picture is the login page which appears after a user submitting wrong passwords several times (six times in author's test).

Moreover, besides the example which is given in Section 1.1, another example is that CAPTCHA will also usually appear when a user is logging into the websites yet input the wrong password several times. In the past, many websites require users to input CAPTCHA when users are logging in; that is for the same reason with the example above. Nevertheless, most websites, such as Facebook, Twitch (Figure 2), prefer not show CAPTCHAs when users give first several tries than show the CAPTCHA every time. Because only a few (below ten) tries from the same user or Internet Protocol (IP) address are acceptable for the server, and this measure makes the user feel more comfortable by typing fewer words. There is another reason for this example use of CAPTCHA: protecting users' account from brute-force attacks.

[5] https://www.twitch.tv

2.1. Types of CAPTCHA

There are three main types of CAPTCHAs i.e. text-based, sound-based and image-based [47]. As shown in Figure 3, picture (a) is the text-based schemes, and (b) is the image-based regimes. In the top of picture (b), the Chinese words mean "Find out and click all the swim rings." Text-based schemes are text images, which usually are static images. This type of CAPTCHA often strongly deforms the characters and adds some noises (e.g. hot pixels, lines as CAPTCHA image (a) in Figure 3) to disturb computer programs' recognition. However, it should be recognizable to human. Sound-based schemes are about the sound recognition task, they may use strange accent and pronunciation of a language and add noise to interfere computer programs' recognition. Image-based schemes are images but can often be dynamic and contain common things such as mountains, lakes, animals, and artworks. As this type of CAPTCHA requires a complicated recognition algorithm, there will not be any noise in the images [19].



(a) (b)

Figure 3 Types of CAPTCHA

Attributing to Text-based schemes' "light weight" and low difficulty to realize, they are the most used kind of CAPTCHA in the world [19]. Text-based schemes can be divided into the dynamic image and static image. A static image is more common than dynamic image. Static image text-based schemes are the image containing some characters (e.g. letters and numbers) with some methods to disturb automatic recognition. Distortion and noises are the common methods to disturb automatic recognition via providing extra information to be processed. Distortion is to change the shape of the character, such as change line to curve and rotate the character in a direction. Noises are hot pixels and lines/curves. For example, Figure 3 (a) is a static image text-based scheme, the slightly twisted characters and some lines are in the picture. Dynamic image text-based schemes are similar to static ones; the differences are the characters and noises may be animated. The text-based schemes will add a question up to the image of "Type the characters in this image below" or "What are the characters in this image below."

There is another type of text-based schemes. It still belongs to the static schemes. However, it requires an extra calculation rather than just recognition. The contents of this kind of text-based scheme are mathematics expression. Thus both of users and robots must do a calculation after recognizing the scheme. For instance, if the content is "88 –

6” or “88 – 6 = ？ ” the users or robots should input “82” as their answers. Since not every people has got enough education, the mathematics expressions may be about the math level of the second grade of primary school, which means it only contains the addition and subtraction within 100. Nevertheless, the difficulty of the mathematics expressions depends on the CAPTCHA’s designer; the high difficulty is also possible.

Sound-based (or audio) schemes just change the words into voice. Like the text-based schemes, sound-based schemes will also have a question in word before that, like “Type the word you hear”. The sound-based schemes are often words or strings of random letters invoice but with some noises and cacophonies. The cacophonies are for disturbing the voice recognition. For example, “X… (noises)…G… (noises)…K” invoice is the sound-based scheme. Furthermore, a simple question can also be the sound-based scheme. “What’s the number after five?” is an example for that [19].

Image-based schemes have more types than other two kinds of CAPTCHA. The basic image-based schemes are like the picture (b) in Figure 3, listing some pictures and asking a user to find out and click all the target items. This type is simple and effective as users do not need to type and just use the mouse to click. Furthermore, few or no distortion and noises will be added to this type of image-based schemes since they will have a grave impact on human’s recognition [19]. Nevertheless, a challenge for this type of CAPTCHA is that the items, sceneries, animals and human characters are not objective comparing with texts. Because the same item in an image may be recognized to different items by users having different educational, cultural backgrounds. For instance, an eastern dragon is more like a snake for western people who never see an eastern dragon. As a result, for some users will seldom get the access and other can easily pass. This means it may cause a situation that 90% of users can pass the CAPTCHA easily (over 95% accuracy) while 10% of users can hardly pass (less than 20% accuracy). Comparing with common situation that all the users will have over 80% accuracy passes, and websites would better not make an exception of some users, the situation caused by this type of image-based scheme needs to be progressed.

There are many types of image-based schemes [20], besides the one which is introduced above, one more type will be introduced. As the same reason, the author cannot list all types of image-based schemes and. Moreover, this thesis will not focus on image-based schemes. Therefore, only one more type to be introduced. It is jigsaw puzzle CAPTCHA. The image of this CAPTCHA can be any image lacking a piece of the fragment (Figure 4). The dragging bar below the image has a button to be clicked and dragged by users. The button can control and move the fragment (with highlight edge) in the horizontal direction. However, there is an extra shadow block just beside the real vacancy of the image. It is a noise in this CAPTCHA. The differences are the value of grey-scale and the small gaps. Users need to drag the button/fragment to the real vacancy so that they can get the access of CAPTCHA.

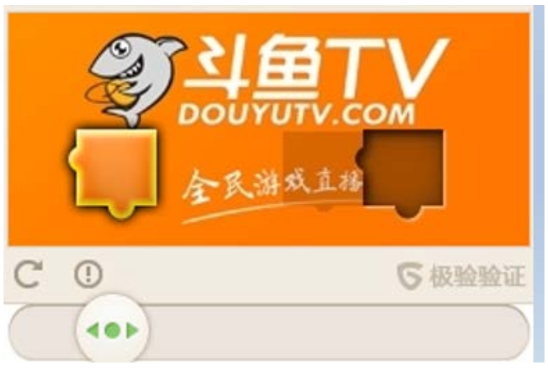


Figure. 4 Example of jigsaw puzzle CAPTCHA

In this thesis, the research will emphasize text-based CAPTCHAs. Furthermore, the CAPTCHAs mentioned in this thesis will be text-based CAPTCHAs unless they are illustrated as other types of CAPTCHAs. There are three reasons. At first, the text-based CAPTCHAs are the most widely used around the world. For example, Microsoft, Facebook, Twitter, their websites use their own text-based CAPTCHAs [19]. Secondly, comparing with other two types of CAPTCHAs, text-based CAPTCHAs have more advantages [21]. For instance, text-based CAPTCHAs cost much less storage space. Therefore, the web page with text-based CAPTCHAs can be loaded faster. Moreover, the ambiguity of text-based CAPTCHAs is lower than sound-based CAPTCHAs and image-based CAPTCHAs because of the fewer localization factors as the image-based schemes paragraph indicated. Furthermore, text-based CAPTCHAs are easier to analyse and realize in technical.

2.2. Usability

Both robustness and usability are non-functional requirements (NFRs) which describe the constraints on a software system. Adler et al. [4] address that usability is to take the best advantage of users’ skills in creating the most effective and productive working environment. Ormeño et al [6] agreed on “usability is a quality attribute related to effectiveness, efficiency and satisfaction of the end-users when they interact with a system.” Both of them considered usability requirements analysis as a challenge.

Usability’s effectiveness, efficiency, and satisfaction is reflected in two aspects: ease of use and elegance and clarity [41]. To be specific, the ease of use includes functional, responsive, ergonomic, convenient and foolproof [41]. Functional is a common factor since it is the core requirement of a product, which means everything of the product will work. A user knows the product is working, and the user is also willing to find out where it is working. Thus, responsive factor requires a just right feedback to improve usability. Ergonomic factor requires that users can easily do the legal operations, including seeing, clicking, poking and turning stuff. Users should feel good wherever they need to go and will go, and everything should be right there, this refers to convenient factor. Foolproof factor requires designers to give correct, clear and straightforward guidelines or guiding directly for users, avoiding them from making mistakes or breaking stuff.

Furthermore, elegance and clarity include visible, understandable, logical, consistent and predictable [41]. Visible factor means that users can see the things of products. Users should be able to know what they are looking at and acquire how it works. Thus the understandable factor is necessary for usability. Logical factor requires everything, such as normal stuff and procedures to make sense if users are seeing it or asking users. Consistent factor suggests that the rules of a product will not change on users unexpectedly. When a user is doing something, the user can have a clear awareness of what will happen next. Therefore, the predictable factor is beneficial to usability.

As the factors above mentioned, the usability is important. In opposite, if the design cannot fulfill the factors well, which is a common fact and risk in the real world [8, 9, 41], the user experience will be destroyed from time to time. For instance, when a man goes on a website just for searching for an interesting video, but he found that he had to register to acquire all information on this website, then he typed over 15 rows of information and clicked submit button. The nightmare happened, there was something wrong with the format of his input information (e.g. email should contain “@”), moreover when the page went back, the user found all the text block was empty! That means he must input the over 15 rows information again. Finally, he just left click the mouse in the top right corner (shut down the browser). In this example, the website’s usability is low because of responsive and convenient. If the website can inform the user that there a mistake in his input information just after the user finishes inputting that row (responsive), the user will hardly make mistakes when he submits the register form. Also, if the website just cleans up the passwords row after the failed submitting, users will be more probably willing to retry instead of exiting everything.

Carroll, M [7] argued that usability in real products was difficult to be analysed academically because analysts who are responsible for system requirements are not experts in this field --- human behaviours or software psychology. For instance, besides the issues from Eric [41] some widely-used usability design guidelines such as Nielsen’s [9] guidelines are also treated as theory-only standards, which means that their guidelines cannot easily be applied to software development directly.

The functional factor for CAPTCHA is determined by CAPTCHA’s definition, easy for human to recognize while cannot be recognized by the machine. As the usability is mentioned here, CAPTCHA should be recognized by human easily is the functional factor for CAPTCHA. The responsive factor is like the example given above; the user can know the result the sooner, the better. The ergonomic factor means “user can easily find where and what to input CAPTCHA answer” to CAPTCHA. As for convenient factor, the user should not be confused and annoyed by the CAPTCHA in any way. The foolproof factor is that users can know what they should do when they meet the CAPTCHA. For example, only a text-based CAPTCHA image and an input box may confuse some users since the requirement is unclear. Therefore, a question like “what’s the characters in this image” is necessary.

From the perspective of the elegance and clarity of a CAPTCHA, visible means that user can see the characters of CAPTCHA. The understandable factor is one of the requirements in CAPTCHA’s definition and a functional factor which is users can know what are in the CAPTCHA. Logical factor asked everything in CAPTCHA should make sense. To be specific, the user should know what noises in the CAPTCHA are. Consistent factor is that CAPTCHA generation program should use only one combination of the designing method. This time shows the CAPTCHA with hot pixels noises and next time shows the CAPTCHA with lines noises will confuse users. Predictable factor means that users can know what will happen after typing CAPTCHA or can predict the rest characters according to the first several characters.

As a result, identifying the risk of usability problems is essential during its development because it will take more time and resources to reduce the problems [11]. Many techniques such as interview, brainstorm, voluntary reporting, decomposition, survey, assumption analysis can be used for identifying risks [12]. Therefore, technology and algorithm analysis are unreasonable for investigating the usability of a product.

2.3. Robustness of CAPTCHA

The definition of Robustness is "the ability of a system to resist change without adapting its initial stable configuration" [13]. It is different from reliability. However, they are similar abilities, just in different environments [14]. Reliability is for the stable environment, and few changes will be made in system configuration. Robustness is to avoid the changes/attacks in system configuration. Chong et al. thought “robustness links confidentiality and integrity properties of a computing system and has been identified as a useful property for characterizing and enforcing security” [15].

Durach, C.F. et al. [16] pointed out that there are two dimensions, avoidance, and resistance, of robustness. Avoidance is defined as the ability of a supply chain not to be affected by change and resistance represents the ability of a supply chain to withstand change [16]. Therefore, Sawik [17] suggested rising resistance to the evolution as a common method. Nevertheless, some changes have to be avoided instead of being resisted.

To start with, the definition of CAPTCHA's robustness will be explained since it is a little confusing. Robustness is an ability that avoiding changes without initial configuration. When the definition comes to CAPTCHA generation algorithms, it means the ability that the algorithm can still generate CAPTCHA when a user (or a robot) types the wrong CAPTCHA many times. However, CAPTCHA's robustness means the resistance to computer programmes written to solve CAPTCHA test automatically. The robustness of CAPTCHA sometimes could also be called "strength" or "safety."

Gao et al. [22] confirmed that "CAPTCHA is now a standard security technology for differentiating between computers and humans." As a result, CAPTCHAs must be robust enough to maintain an acceptable level of safety. Also, many types of research about cracking CAPTCHAs were proposed. Therefore, the robustness of CAPTCHAs faces a tough challenge.

To quantize the robustness of CAPTCHA, attackers' perspective is necessary since robustness is oriented to automatic apps. From attackers' perspective, three factors will be considered at first: "How advanced are techniques being required?" "How much time will the recognition process cost" and "Are the predictions of recognition correct?" which mean the technical difficulties, processing time and accuracy [22]. However, the technical difficulties cannot be quantized directly, while the other two factors are more intuitive in number. As a result, the recognition processing time and prediction accuracy can quantize the robustness of CAPTCHA from attackers' perspective.

The processing time can be divided into several parts according to the specific recognition process, which will be introduced in Chapter 3. The longer processing time is, the more time the attackers will spend on validation and testing algorithms. The prediction accuracy means how correct the results of recognition will be. A high accuracy can ensure an acceptable cracking result. In opposite, low accuracy will resist the automatic recognition. Furthermore, the two factors can be considered with other elements. For instance, in the robustness test made by Gao et al. [22], they used the accuracy of different classifier algorithms and the amount of training data as their robustness evaluation standard. The amount of training data can determine the directly: the most training data collected, the higher accuracy will be. However, the word “training data” is from machine learning part, which will be introduced later (Section 3.3). As a result, the metrics of the robustness of CAPTCHA are processing time and accuracy of prediction. Moreover, these two factors, especially the accuracy, will be used for the robustness test/ recognition process.

A computer programme can remove the noises and distortion so that it can acquire the information in CAPTCHA more correctly in automatic. However, the removal will not be smooth since some noises and distortion cannot be removed without any cost. To be specific, firstly, the removal process will cost processing time and require attacker’s ability level. Then, the removal can be inaccurate, which means the wrong recognition will cost more time on the attempts. At third, to make the programme automated, some algorithms will be used to teach a computer to learn the information and transform it to the answer, the process will also cost time and may give wrong answers.

2.4. Factors Affecting the Usability of CAPTCHA

CAPTCHA is a program that generates and grades tests that are human solvable, but beyond the capabilities of current computer programs [18]. As a result, the low usability or unusable CAPTCHAs should not exist. However, in fact, some CAPTCHAs did not do well in usability since it is necessary to keep a high level of robustness.

Yan and Ahmad [19] summarized the usability issues with text-based CAPTCHAs, as shown in Table 1. There are three categories: distortion, content and presentation.

Category	Usability issue	
Distortion	Distortion method and level	
	Confusing characters	
	Friendly to foreigners?	
Content	Character set	
	String length	How long?
		Predictable or not?
	Random string or dictionary word?	
	Offensive word	
Presentation	Font type and size	
	Image size	
	Use of colour	
	Integration with web pages	

Table. 1 Usability issues with text-based CAPTCHAs [19]

2.4.1. Distortion

Distortion is the alteration of the original shape of something, such as an image, sound and waveform. As for text-based CAPTCHA's distortion, some or all characters in the CAPTCHA image will be distorted in a way such as twisting, slightly inclining and other distortion methods. Twisting is to change the character structure, for example, amend the straight line into curves or alter the radian of some curves. Slightly inclining is not to changing the characters directly, it just turns them in a small angle. As common humans would feel difficult to recognize the distorted characters in text-based CAPTCHA, the distortion will affect the usability of CAPTCHA clearly.

Distortion method and level. Readability is a prerequisite of CAPTCHA. Thus, it is essential to CAPTCHA's usability. However, the distortion methods and the level of distortion can determine the readability. Four common distortion methods for CAPTCHA were examined by a team [21]:

- *Translation*: move the characters to any directions by amount
- *Rotation*: turn the characters by a radian/angle clockwise or anticlockwise
- *Scaling*: enlarge or shrink the characters in coordinate axis
- *Warp*: change the shape of CAPTCHA images elastically

As a result, distortion is concerned with both of the ease of use and elegance and clarity. To be specific, distortion level and method may change the functional and ergonomic factors in ease of use since it can determine the readability. Readability is the basic functional attribute of CAPTCHA. Therefore, distortion level and method may change the functional factor. Furthermore, the warp method may influence the ergonomic factor thanks to the impact on the user interface appearance.

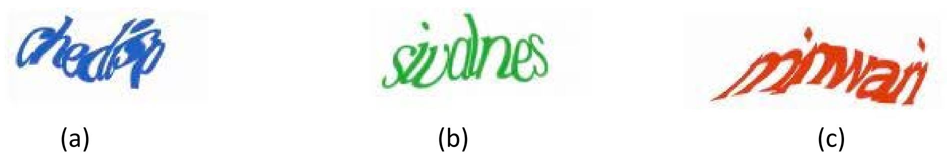


Figure. 5 Examples of distortion CAPTCHA images

In Figure 5, the three pictures are examples of distortion CAPTCHA images using warp, translation and scaling as their distortion methods. The colour picture (a) is blue, (b) is green, and (c) is red.

Confusing characters. In normal CAPTCHA, which is not distorted, there may be some ambiguity of some characters. For example, the letter "o", "O" and the number "0"; the lower-case character "i", "l", the capital character "I" and the number "1" are common confusing characters because they are too similar in some way to distinguish. Moreover, distortion may enhance the ambiguity even more. For instance, in the blue CAPTCHA (a) of Figure 5, the first two characters can be recognized as “dn” or “ch”, and the “d” in the middle, it can be recognized as “d” or “cl” . In addition, besides the functional factors, confusing characters may be concerned with the ease of use since it can decrease the feeling of a convenient factor. Users will not feel right when they meet the confusing characters.

Friendly to foreigners. As the text-based CAPTCHAs is so popular, which is mentioned in Section 2.1, they are often used in international oriented websites. Therefore, text-based CAPTCHAs have an advantage that little localization issues need to be found in them [30]. Nevertheless, some people who are not from the countries which use the Latin alphabet (e.g. English) may still be confused with the distorted letters. Yan and Ahmad [19] did an experiment, confirming that only a small difference between natives (people from the US) and foreigners (users from Asia) when recognizing the distorted text-based CAPTCHAs using English (accuracy rate 97% for natives and 93% for foreigners). However, the small difference may lead to a serious usability problem when a large quantity users using the service, especially world-spread websites, such as Facebook and Twitter.

As a result, using English alphabets as the CAPTCHA’s characters is acceptable for most situations. However, if the localization of language can be done automatically, the ease of use will be enhanced by the convenient factor is improved. It is the bonus effect which can be ignored for little loss while can still gain for the improvements.

2.4.2. Content

Content, as the word suggests, is the content embedded in CAPTCHA. There are four issues describing CAPTCHA’s content: character set, string length, random string or dictionary word, and offensive word.

Character set. The size of CAPTCHA’s character set is concerned with its robustness. The larger character set is, the tougher cracking CAPTCHA is. Nevertheless, a bigger character set can also cause confusion for implying more characters that look similar when they are distorted. Moreover, it is still concerned with the ease of use for the convenient factor. This is similar to confusing words and has less effect on usability.

String length. It is also a matter for robustness. In a common situation, the longer string used in CAPTCHA, the more difficulties will be in automatic recognition. As for the impact on usability, it is interesting that string length will subtly influence usability in a different way in the different situation. For instance, if the string is using random words, it is sure that the longer, the more annoying. It will decrease the usability by not only decreasing the recognition accuracy (with distortion of CAPTCHA) but also annoying user for typing too much. However, if the string is using the English words, the accuracy will be increased with longer string since the user can get more information from the known characters. This factor is concerned with convenient in ease of use; people would fell impatient when they should type a long length string.

Random string or dictionary word. As it mentioned in string length, random string and dictionary word will influence the usability in many ways with string length and character set. Nevertheless, it can be found that lexical information has already been used in automatic CAPTCHA recognition [31]. Therefore, using a dictionary word as the string may be a risk of CAPTCHA's robustness. As a result, using the word that not in dictionary while can give other hints to human users is a good method to improve the usability while keeping the robustness. For example, creating a non-English but pronounceable string, it may be difficult for the moment. However, it is potential and feasible. Thus, random string or dictionary word is connected to predictable in elegance and clarity.

Offensive word. This issue is not as common as the issue above for the usability. Regardless, it does impact on usability for some situation. For instance, show the offensive word, such as "Negro". This situation can appear in both random words and dictionary words. As a result, this factor is concerned with convenient in ease of use. People will be annoyed by the offensive words.

2.4.3. Presentation

Presentation decides the ways that show the CAPTCHA, for example, the font type and size, the image size, the difference colour and Integration with web pages.

Font type and size. This is the matter of robustness, a different type of font can determine the basic recognition methods of cracking program. For example, matrix matching method is suitable for the common typewritten text, and feature extraction performs better in handwritten text. On the other hand, different font types and sizes will change the usability. For instance, some handwritten text (e.g. curlicue) is really difficult for the users whose mother tongue is not English. Moreover, too large or too small size will force users to find out the characters features very carefully. This unpleasent element will annoy users if users are doing something in a hurry or something unimportant. Thus, font type and size are concerned with visible and understandable in elegance and clarity. Too small size characters make user find difficult to see them. Moreover, some special font type may be complicated to understand.

Image size. It is related to the websites user interface design. A large image size of CAPTCHA is unnecessary and cost too much space so that the layout of the web page will be harder. In opposite, a small image size of CAPTCHA is easy to deal with in user interface design. However, if the CAPTCHA size is too small, the user may not find the input bar or feel annoyed to be forced to recognize CAPTCHA carefully in a small image size. Thus, the image size is connected to ergonomic in ease of use and visible in elegance and clarity since the image size will have an impact on UI design.

Use of colour. Colour has some advantages in usability, for instance, colour can improve the design of user interface. There are five reasons for using colour in text-based CAPTCHA in common [19]:

"

- Colour can attract people directly and effectively.
- Colour can fulfil different user preferences by its variation.
- Colour can make more fun CAPTCHA challenges.
- Colour can help with recognition, understanding and positive affect.
- Colour can make CAPTCHA images' compatibility better via the colour of web pages and make them look less annoying. "

However, many uses of colour in CAPTCHA cannot help improve usability, and even have an adverse impact on both robustness and usability [19]. As the Figure 6 indicates, the grey scale of background is obviously lower than the characters because characters are black, which is the maximum grey scale. In this situation, the background can easily be removed by a computer, this is no use for improving robustness. In addition, the distorted background enhances the CAPTCHA's distortion which will decrease the usability. Thus, the use of colour is concerned with convenient in ease of use and visible in elegance and clarity.

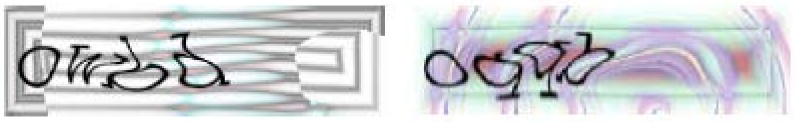


Figure. 6 the useless utility of colour in CAPTCHA

Integration with web pages. It is also concerned with usability. For example in reCAPTCHA (Figure 7), the "type the two words" up to the input box in the scheme will not appear. First, that means clicking the CAPTCHA is needed for showing the hint words. However, it will decrease the usability of CAPTCHA via increasing the burden of users. To be concrete, users must activate the hint words before they can type the answer. To avoid that, CAPTCHA and a web page can be an integrity so that the burden will be less.

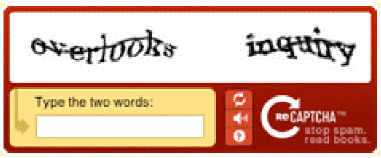


Figure. 7 Example of reCAPTCHA

Ho et al. [20] tried to make a game for evaluating the usability of CAPTCHAs (Figure 8). He pointed out that "some enhancement procedures make the CAPTCHAs too difficult to be recognized by human, for example, with too noisy background or too much text distortion". As a result, he found that a CAPTCHA, which has the good usability should be recognized by human easily, quickly and not annoyingly. The standard should be quantized to evaluate the usability of CAPTCHA. However, only "quickly" can be quantized directly. "Easily and annoyingly" are about human's feeling. Therefore, they are unquantifiable. Ho et al. [20] presented good methods to evaluate "easily and annoyingly". A game was created for assessing the usability of CAPTCHA, and five metrics are taken into account. They are finish time, the rate of a typing error, the rate of the timeout, the rate of giving up, the rate of repeat typing.

Finish time is the total time to solve the CAPTCHA. This quantizes the human's processing time and the difficulties of initial impression recognition. The rate of typing error is the number of the errors that players type the wrong words of CAPTCHAs during all game; then the number will be divided according to the number of CAPTCHAs. In a word, it is the accuracy of typing CAPTCHAs words. The rate of timeout is calculated by dividing the number of timeout CAPTCHAs by the number of all CAPTCHAs. The limitation of time is three seconds; timeout means the user did not finish typing the CAPTCHA in three seconds after choosing a CAPTCHA as his target. The rate of giving up is to divide the number of CAPTCHAs given up by the number of all CAPTCHAs. If the player chooses a monster (with CAPTCHA) as his target, he can press the SPACE key to leave the choice before the timeout. The rate of repeat typing means the number that counting the repeat keypresses; it can be divided by the number of all keypresses. Players can enter the same character to prevent a timeout.

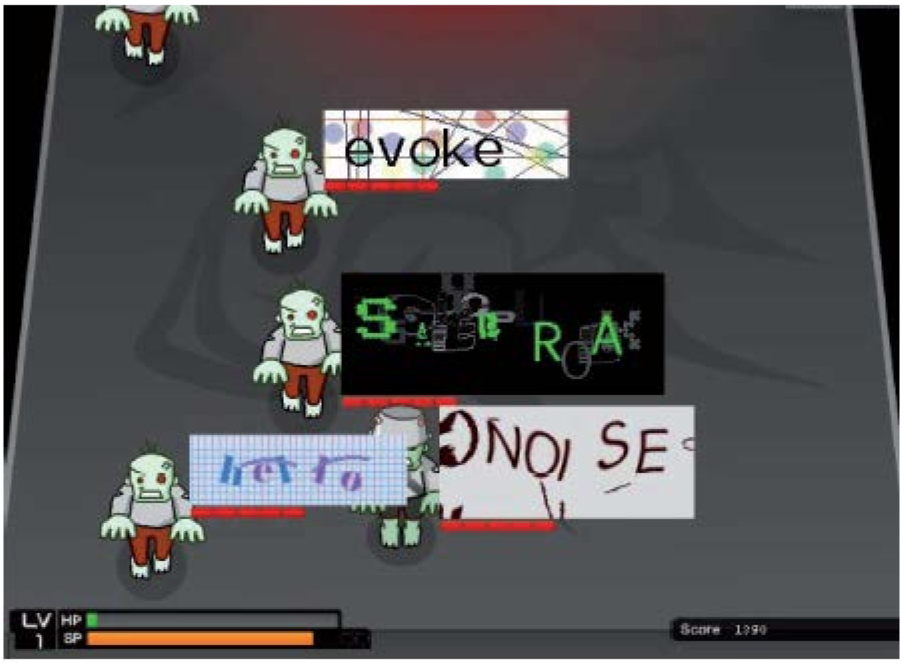


Figure. 8 a screenshot of the game

There was also research on colour's impact on CAPTCHA's usability, written by Ahmad et al. [23]. They proposed that colour could easily improve the usability while made only a few changes of robustness because the colour is a usability issue. However, some colours or uses of colours may still impact the robustness of CAPTCHAs. For example, if the colour of image's background is too close to the colour of text part of the image, this will decrease the usability, furthermore, have limited impact on the increase of robustness.

2.5. Factors Affecting the Robustness of CAPTCHA

As it mentioned in Section 2.4, CAPTCHA's distortion, contents (character set, string length, and dictionary word), font types and colour are concerned with CAPTCHA's robustness [19].

Distortion is mainly interfering the digital image processing part. For example, adding noises (pixels or lines) to the CAPTCHA image; conglutinating characters in the CAPTCHA together to make segmentation more difficult. Therefore, distortion can impact on the robustness directly: the more distortion is made in an image, the more difficult pre-processing will be.

Contents will affect CAPTCHA's robustness by increasing the information, such as matrix data and features database, which the recognition process needs to learn. The large character set will obviously enhance the quantity of training data for keeping an acceptable accuracy.

A long string will increase the processing time on recognition. However, only being dramatic on the image processing and recognition process using lazy-learning classifier. If the recognition process is using model-based learning methods, such as decision tree, the increase of robustness will be limited.

As for the dictionary words, it can only affect a little on robustness since this thesis is going to make specific methods to test robustness. A dictionary detecting function will be made according to the specific kind of CAPTCHA. In this situation, the enhancing of CAPTCHA's robustness will be limited.

Font types work in a similar way as the character set. It can increase the robustness of CAPTCHA by increasing the quantity of training data. However, this issue may perform much better if the characters are using random font types in a type font set. For example, a string "abcd", "a" is the "Times New Roman" font, "b" uses curlicue, then "c" and "d" use "Microsoft YaHei". In this situation, the number of training data will increase in geometric growth.

However, as they are also concerned with the usability, it is better to do an experiment for testing these elements for calculating the levels of improving the robustness of CAPTCHA. Then the results will be easier to be compared with the usability issues.

In addition, three guidelines of uses of colours are already concluded by Ahmad et al. [23]:

1). "Uniformity in the foreground or background reduces resistance to segmentation attacks", therefore, use different colours in different characters or add noises in background (black and white images) is good principle for robustness;

2). "Contrast between foreground and background, or contrast among foreground characters, also reduces resistance to segmentation attacks", as a result, using colours which enhance the contrast will reduce the robustness;

3)." Perceptually connected but physically disconnected components are another good security design principle." Nevertheless, the perceptual grouping may be increased or decreased by using colour. Therefore, an evaluation for CAPTCHA's design is necessary.

As a summary, colouring is a double-edged sword for CAPTCHAs; complex colour schemes might enhance the robustness while with a decrease of usability. As a result, many websites such as Microsoft, Google use simple colour schemes now [23].

Nevertheless, the research on both usability and robustness simultaneously still need to progress. Furthermore, it is better to use the experiment to test the robustness, which will be given in Chapter 4.

3. CAPTCHA Recognition Methods and Techniques

3.1. Methods to Automatically Recognize CAPTCHA

As the research method includes cracking CAPTCHAs for examining the robustness, CAPTCHAs' recognition is an advanced technology, and the methods that automatically recognize CAPTCHAs are explained in this chapter.

In the digital image recognition area, Optical Character Recognition (OCR) is considered as the most popular technique. It is a common method which is combined with digital image processing and machine learning to recognize digital images automatically. As the CAPTCHAs in this article are text-based schemes, OCR is certainly an ideal method to recognize CAPTCHAs automatically.

Before the recognition, all CAPTCHA images need to be pre-processed. There are a few steps in common: de-noising (noise removal), segmentation and identifiable data generation. For colour images and grey-scale images, the binarization will be added before de-noising. These will be introduced in Section 3.2.

Optical character recognition (OCR) can transform the words into images to the matrix with black and white pixel via optical methods, moreover, to text-based files. To evaluate OCR, percentages of reject rate and error rate, and speed of recognition are taken into account.

OCR contains the pre-processing steps, denoising and segmentation and normalizes aspect ratio and identifiable data generation. Differences are created by the specific algorithms. Two basic character recognition methods are used in OCR in common: Matrix matching (pattern matching) and Feature extraction [26]. Matrix matching is to transform the image into a matrix data, stored pixel-by-pixel. This measure performs the best with typewritten text, and only recognizes the fonts it stored. Feature extraction is to extract the features from images, such as descriptions, values, and vectors. The characters can be recognized by comparing with their extracted features. This method is suitable for handwriting text and has the widest applicability in modern OCR applications. Therefore, the recognition method should be determined according to the actual situation.

However, OCR is a generic method, which means it can deal with various text-based CAPTCHAs, and has a lower accuracy [28]. Therefore, a specific recognition algorithm needs to be developed for specific kinds of CAPTCHAs to reach higher accuracy. The research method of OCR is still worth learning.

OCR provides a generic measure for recognizing digital images. However, there are many specific algorithms to realize the steps of OCR. Vector space model is one of them. In addition, Vector Space Model (VSM) was proved that could implement a Vector Space Image Recognizer (VSIR) to recognize CAPTCHAs efficiently [24].

A VSIR (Vector space image recognizer) is defined by Wong et al. [27] that it is "essentially an application of the VSM, where the stored entities are compared with each other or with incoming search requests." This is achieved by modelling the various information retrieval objects as elements of a vector space and by employing matrix analysis techniques to find the relations and key features in the entities. [27]

The VSIR cannot become over-trained. As a result, increasing the amount of training data can rise the accuracy. However, it also cost more time on classification process. When new training data (CAPTCHA images) are being added, all the rest data have to be indexed again and more time is spent on an additional training. VSIR also requires a pre-decided method because it cannot calculate a new solution by itself like neural networks.

As a conclusion, to analyse the factors which can influence the robustness of CAPTCHA, the basic steps of OCR will be taken into account, i.e. digital image processing, and machine learning. Moreover, specific algorithms for machine learning part will be discussed in details. Although there are many other algorithms to break CAPTCHA, a specific algorithm will be selected according to the schemes of CAPTCHA to get a high success rate. Furthermore, unlike the progressed algorithms such as VSIR, basic techniques and algorithms can help to understand the process of automatic CAPTCHA recognition better. As a result, some basic techniques and algorithms will be introduced next.

3.2. Digital Image Process

As mentioned in Section 3.1, the pre-process is mainly the digital image processing, which means turning raw images to the numeric and easy-to-handle data. As the thesis focuses on the machine learning, the pre-process of CAPTCHA recognition, a phase before the machine learning phase, is briefly introduced in this Section.

Regarding the pre-processing, as it is not the emphasis of the thesis, the basic tasks of pre-processing will be introduced here. They are denoise, sharpen/smooth, segmentation and data transforming. The first three tasks are mainly to remove the noise in the training data. As machine learning part cannot a success if there exist noises in training data, the three tasks are necessary for every automatic CAPTCHA recognition programme. The last task is to transform image data to a treatable structured (e.g. CSV files, digital matrix) or semi-structured data (e.g. JSON files) for machine learning.

De-noise is the process that can remove most obvious noises (e.g. hot pixels and lines). The process is initial and will still leave some noises. Furthermore, it sometimes may also remove the useful pixels. Therefore, the sharpen/smooth process is needed to eliminate the noises around the target Figure, which are often left after being de-noised. The sharpen/smooth process can also fix some mistakes made by de-noise, for instance, some edge pixels of the Figure may be removed by de-noise too, and some new noises may be created in the inner parts of the Figure. After the image being applied to sharpen/smooth process, the most noises (over 90%) should be removed.

De-noise is one of the primary use of both linear and nonlinear image in image enhancement [39]. Applying a filter operator to a pixel and its neighbourhoods is the common methods to de-noise and smooth. The filter is a common method to process digital image, filter operator is usually a matrix. To enhance the image, every value of filter operator will be timed to a pixel and its neighbourhoods, then the value of pixel (grey-scale) will be replaced by the value of the filter. Thus, it will smooth the image via making the pixel more similar to its neighbourhoods [39]. The neighbourhoods of a pixel are typical, expect the edge pixels, all other pixels from eight different directions around the pixel.

Then the segmentation will be applied. The noises can also disturb the segmentation part. Thus, segmentation will be added after de-noise and smooth. Segmentation can split the target Figure into several individual Figures. For example, if the target Figure is a string, the segmented Figures will be the individual characters. Finally, it is the data transforming; the process is to transform image data into digital structured/semi-structured data. Specifically, the data transforming will be used in the thesis is to transform processed images into matrices.

The basic purpose of segmentation is to “partition the image into mutually exclusive regions to which we can subsequently attach meaningful labels.”[39]. Correct results of segmentation will be the object or region in which users are interested. Two basic routes of segmentation are concluded by Chris and Toby [39]: “

- Edge/boundary methods: This approach is based on the detection of edges as a means to identifying the boundary between regions. As such, it looks for sharp differences between groups of pixels.

- Region-based methods: This approach assigns pixels to a given region based on their degree of mutual similarity. ”

There are also some complementing tasks, such as binarization and rotation. Binarization is only used for colour images and grey-scale images. It can enhance the useful part and whiten the useless part. This can help the segmentation since binarisation makes the edges of each character obvious. Rotation is to straighten all the rotated characters, but it is also unnecessary because the rotated characters cannot create noises directly, it can also be solved with a broad diversity of training data. As this part is not the thesis emphasis, the unnecessary tasks will not be introduced more.

After the segmentation and other complementing tasks, the different segmented images of characters will be given. Moreover, these images should be binary image and formed into the size as users’ wish. Therefore, the binary matrix generation can be handled by a bitmap segment---just bijection between the segmented bitmap image and the binary matrix.

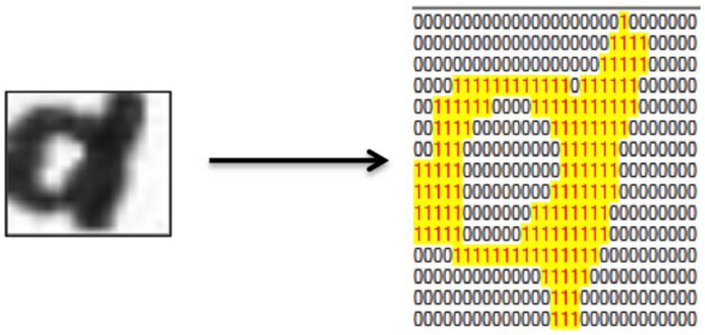


Figure. 9 binary matrix generation

3.3. Machine Learning

After getting enough binary matrix data, the recognition step can be started. Machine learning is to teach a computer to give some reactions automatically. Thus, the basic steps of a machine learning are collecting data, cleaning data, applying learning algorithms to the data, then the computer can learn to do some reactions according to the learning algorithms.

There are some types of learning algorithms. For example, supervised and unsupervised algorithms; lazy-learning and eager-learning algorithms [35].

In supervised learning. Each line of the training data has their class. To be specific, a researcher can clearly know to which class the attributes belong. For example, in the generated matrix in Figure 9, each line has its class, e.g. [0, 0, 0…1...0, d], the “d” is the class. It means the attributes [0, 0, 0…1...0] belongs to class “d”.

Unsupervised learning. The class information is not available which means the training data does not have the classes, and each line of data does not belong to anything for the moment. Therefore, in cluster analysis, the classes (clusters) will be formed in this kind of methods. K-means clustering is the most popular basic ways in clustering methods [32].

In lazy-learning algorithms, computation will keep waiting until a value for a new case is predicted. To be specific, the learning process will be applied every time that a new case needs to be classified or predicted, because this kind of algorithms will not build a model or features set to make the prediction. Therefore, the learning phase will be faster, and prediction or recognition phase will be slower. K-nearest neighbour is one of the typical algorithms of lazy-learning.

Eager-learning is the opposite of lazy-learning. Eager-learning algorithms will build a model for the training data. Then the model will be applied to predict values for new cases. Therefore, the learning phase will be slower, and prediction or recognition phase will be faster. A decision tree is one of the typical algorithms of eager-learning.

A learning method can belong to different types, for instance, the k-nearest neighbour classifier is the method that is both supervised learning and the lazy-learning. These four basic learning algorithms will be introduced in details.

3.3.1. K-means clustering

K-means Clustering is the unsupervised learning method and belong to clustering analysis whose aim is to find groups of objects (clusters) such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups [34].

The process of k-means is, first, select the value of K, this means the number of clusters. Therefore, K is the number of clusters user wants to be. Secondly, choose K points randomly as the initial cluster centres, and the first K areas are decided. Thirdly, calculate the Euclidean distance[6] between each case and a cluster, and assign the cases to a cluster

[6] For points or cases $x(x_1, x_2,\dots x_n)$ and $y(y_1, y_2,\dots y_n)$:

$$\text{Euclidean distance} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

which is the closest to the case. Compute for each cluster the mean vector[7] of the points allocated to the cluster. Then use these mean vectors as new cluster centres. Repeat the third process until there are no changes in clusters. The pseudocode of k-means algorithm [32]:

```
1: First fix the number of clusters K.
2: Arbitrarily pick K points as initial cluster centres.
3: repeat:
4:     Assign each case into a cluster whose centre is closest to the case in the
       Euclidean distance sense.
5:     Compute for each cluster the mean vector of the points assigned to the cluster.
       Use these mean vectors as new cluster centres.
6: until There are no changes in clusters.
```

Therefore, the time complexity of k-means can be calculated as O (n· K· I) for n = number of points, K = number of clusters; I = number of iterations [37].

Nevertheless, not all values of K can lead to a high quality of clustering. The quality of clustering can be validated by the sum of squared error (SSE) [32]:
1). For each point, the error is the distance to the nearest cluster centre
2). SSE is the total sum of the squared errors.
3). x is a data point in cluster C_k and I_k is the centre for cluster C_k
4). Given two clusterings, the one with the smallest squared error is preferred.

$$\text{SSE} = \sum_{k=1}^{K} \sum_{x \in C_k} d(r_k, x)^2$$

Formula of calculating SSE [32]

With the help of formula of calculating SSE, a K value selecting method could be found. First, set K = 2, then do the k-means process until there are no changes in clusters. After that, calculate the SSE of this clustering and save its value. Next, increase K by 1, which means K = 3 this time. Do the k-means process again and calculate its SSE. Loop this step with increasing K by one until K reach half of the number of the cases. Finally, compare and sort the list of values of SSE, choose the smallest one as the best K.

[7] For cluster's points $x1(x1_1, x1_2, ...x1_n) \ldots xm(xm_1, xm_2, ...xm_n)$

$$\text{Mean vector} = \left(\frac{\sum_{i=1}^{m} xi_1}{m}, \ldots \frac{\sum_{i=1}^{m} xi_n}{m}\right)$$

However, there is some limitation of k-means. For instance, there may be some problems for k-means when clusters are of differing sizes, differing densities or non-spherical shapes. Furthermore, as k-means is an unsupervised method and the CAPTCHA's matrix data have the clear classes, it is not reasonable to apply it to the CAPTCHA recognition in this thesis.

Fortunately, k-means clustering is quite a good method for the image processing part [40], it can still be utilized in the custom algorithm. Clustering is not only available for data but also utilizable for points in the graphic. The attributes of the data are equivalent to the points' coordinates. It can cluster the points nearby according to the value of k. As the number of characters is determined in a CAPTCHA, the value of k is easy to select. If there are six characters in the CAPTCHA image, the k will be 6, which means there are 6 clusters to be clustered. As a result, apply the points' coordinates to k-means clustering will segment the characters in the image.

3.3.2. K-nearest neighbour classifier

K-nearest neighbour classifier is the simplest instance-based method [34], a supervised and lazy-learning method. Therefore, it is used for the training data that has class labels, and it can perform well in the training process.

The instance-based method is for the training cases (or a subset of them) that are stored during the learning phase, which means learning is storing of training data. A generalisation which is over the learning data will be waiting until the prediction for a new case is done. Finally, to predict a value for a new case, it will search the training data for a case which is similar/near to the new case. The training case gives an estimate for the asked value. The value to be predicted may be quantitative (prediction) or qualitative (classification).

As a result, k-nearest neighbour's core idea is based on instance-based method, storing the training data, then finding the nearest neighbour from the training data of the target cases and classifying it to its nearest neighbour. To find the nearest, the definition of the nearest and methods of calculating distance should be indicated. Here are the steps of k-nearest neighbour:

1). Training algorithm:

- Storing example cases

2). Classification algorithm:

- Search for the new case x its k-nearest neighbours.
- Set as the class of the new case the majority class of its nearest neighbours.

Classification algorithm's pseudocode [49]:

```
Let k be the number of nearest neighbours and D be the set of training data.
    for each new case $x = (x_A, x_C)$ do
        Compute d ($x_A$, $y_A$), the distance between
        $x$ and every example $y = (y_A, y_C) \in D$
        Select $D_x \subseteq D$, forming a set of k that contains all the nearest neighbour
        $x_c = \arg\max_{v} \sum_{y = (y_{A_i}, y_{C_i}) \in D_x} I(v = y_{C_i})$ [8]
    end for
```

Where x_A is the attributes of the case x, x_C is the class of the case x, v is a class label, y_{C_i} is the class of the nearest neighbour i and I () is an indicator function, it will return value 1 when the argument is true, in opposite, it will return 0 when the argument is false [49]. As a result, the time complexity of k-nearest neighbour is O(k · n) for n is the number of lines of training data cases (D) and k is the number of nearest neighbour [49].

As the algorithm demonstrates, the algorithms for computing distances have not been undecided yet. Nevertheless, to find out the distance computing measures, some questions need to be solved at first.

As for the definition of distance between cases, it can be solved by the help of proximity measures that have been calculated by the properties of cases: Similarity, the most similar values of attributes are, the closer they are to each other. Distance/Dissimilarity, the larger disparity of values of attributes, the longer distance, are between the two cases. The term proximity is frequently used as a general term to refer to a similarity or dissimilarity measure. Other terms often used in the same context are distance and metric.

The term distance is often used informally to refer to some dissimilarity measure, and a metric is a dissimilarity measure that fulfils the following conditions [49]:

1) Distance (x, y) ≥ 0 for all x and y. For distances cannot be less than zero.

   And distance(x, y) = 0 if and only if x = y. For an object has no distance to itself.
2) distance (x, y) = distance (y, x) for all x and y. For its symmetry.
3) distance (x, y) ≤distance (x, z) + distance (z, y) for all x, y and z. For triangle inequality: the sum length of any two edges of a triangle will always larger than the length of the other edge.

[8] Majority voting

Then comes to the distance measures that is suitable for quantitative data.
Manhattan or city-block distance:

$$\mathrm{M(x,y)} = \sum_{a=1}^{m} |x_a - y_a| \qquad [32]$$

Where **a** is an attribute, m is the number of attributes, x_a is the value of the attribute **a** for the case x, y_a is the value of the attribute a for the case y.

Euclidean distance is also mentioned in k-means clustering.

$$\mathrm{E(x,y)} = \sqrt{\sum_{a=1}^{m} (x_a - y_a)^2} \qquad [32]$$

The example for indicating the process of calculating distance via different distance measures is given in Figure 10.

Case	Att1	Att2	Att3	Att4	Att5
Case A	1	5	7	3	6
Case B	1	6	4	0	9

$$\boldsymbol{M(A,B) = |1-1| + |5-6| + |7-4| + |3-0| + |6-9| = 10}$$

$$\boldsymbol{E\ (A,B) = \sqrt{(1-1)^2 + (5-6)^2 + (7-4)^2 + (3-0)^2 + (6-9)^2} = \sqrt{28} \approx 5.29}$$

Figure. 10 example of different distance measures

The preceding measures assume that the attributes are commensurable to some degree. Often attributes are not commensurable, but they have been measured using different units. The choice of the unit affects to the contribution of the attribute to the distance measure. A common strategy to the problem of incommensurability is to standardise (or normalise) the data e.g. by dividing each attribute by its sample standard deviation or range. After standardisation, attributes are considered as equally important.

As for the binary data, the similarity measures are more reasonable for them. Let x and y be cases measured by binary attributes (values 0 and 1). Similarity measures are based on the following quantities: At first, suppose the parameters n_{00}, n_{01}, n_{10}, n_{11} to be defined as below [48]:
n_{00} = the number of attributes where x is 0 and y is 0
n_{01} = the number of attributes where x is 0 and y is 1
n_{10} = the number of attributes where x is 1 and y is 0
n_{11} = the number of attributes where x is 1 and y is 1

On this basis, there are three methods for evaluating the similarity:

1). The simple matching coefficient:

$$\mathrm{SMC}(x,y) = \frac{n_{00} + n_{11}}{n_{00} + n_{01} + n_{10} + n_{11}} \quad [32]$$

It is the common method and has wide applicability for all binary data.

2). The Jaccard similarity coefficient, which is the number of 11 matches divided by the number of attributes not involved in 00 matches.

$$sim_{Jaccard} = \frac{n_{11}}{n_{11} + n_{10} + n_{01}} \quad [32]$$

Attributes may describe the presence (1) or absence (0) of certain properties. We may consider that the absence of certain properties from the both cases is irrelevant: The 00 cell is excluded when calculating the similarity.

3). The Dice similarity coefficient, which is the Jaccard coefficient, but the number of unmatched is divided by 2.

$$sim_{Dice} = \frac{2n_{11}}{2n_{11} + n_{10} + n_{01}} \quad [32]$$

If the 00 matches are irrelevant, the unmatched 01 and ten are considered to be between the 11 and 00 matches of the relevance. The number of unmatched is divided by 2 for making an emphasis on what matters (e.g. 1 means the pixel point in an image and 0 means blank).

The similarity can also be applied to quantitative data, and there are more methods for the mixed type data. However, the data will be used in this thesis is binary data. Therefore, the methods introduced above is enough for this thesis.

As the k-nearest neighbour's process and calculating methods are demonstrated, here comes the k's value selecting. Since k is concerned with this algorithms' time complexity, it is essential to have a method to determine the value k.

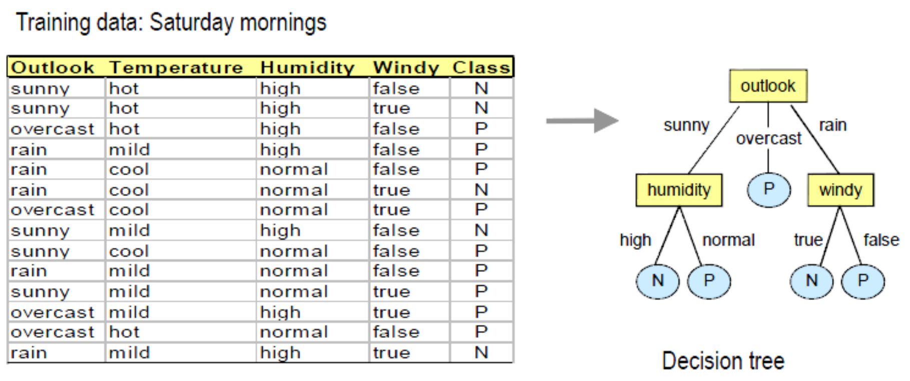
Nevertheless, for selecting the best value k, a validation method should be used as an evaluation method. Unlike k-means, which has evaluation parameter SSE, k-nearest neighbour has to use validation methods to test its accuracy of the predictions/recognitions. As the validation will be introduced in the following section, the selecting method will also be introduced there.

As a summary, k-nearest neighbour is reasonable for the data that being processed from CAPTCHA and used in this thesis.

3.3.3. Decision tree

A decision tree is a supervised and eager-learning method. Therefore, it is utilized for the training data that has class labels, and it can perform well in prediction/recognition process. It is also a kind of inductive learning, which can deal with general knowledge from separate cases.

Knowledge is represented in the form a decision tree, and a classification model to do the prediction/recognition.


Training data: Saturday mornings

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

Figure. 11 example of decision tree [37]

In Figure 11, in the class label, the attribute P means “play tennis”, N means “don’t play tennis”.

Inner nodes include the tests which are formed by their attributes (test nodes). Branches correspond to the outcomes of the tests (attribute values). Leaf nodes (leaves) contain the class information (one class or class distribution).

When classifying a new case, it is started from the root of the tree. Then the value of the attribute will determine which next branch is to go. The branch will lead to a new node (attribute). Therefore, the prediction goes to the next node according to the value of root. The process will not stop unless the leaf node has no children leaf/leaves. At last, the leaf presents the prediction to the class of the new case.

The classification approach from the root to a leaf will give an explanation for the decision. According to the classification path, the number of the tested attributes will be determined. Moreover, testing all the nodes in all paths is unnecessary. A classification path means a conjunction of constraints which is set on attributes and a decision tree stands for a disjunction of the classification paths.

To be general, the only basic algorithm of decision tree will be introduced in details; other algorithms will just be mentioned roughly. Also, most algorithms are based on the basic algorithm TDIDT (Top-Down Induction of Decision Trees).

TDIDT is for a decision tree that is constructed in a top-down (from the root to the leaves), recursive, divide-and-conquer and greedy manner [37]. At first, all the training examples will be at the root. If the argument (stopping criterion) is satisfied, a leaf node will be formed. Otherwise, the best attribute is selected according to some criterion (a greedy algorithm), and a test node is formed, cases are divided into subsets via being based on the values of the chosen attribute. Moreover, a decision tree is formed recursively for each subgroup. The pseudocode of creating a decision tree using TDIDT [37]:

```
(1)  Create a node N
(2)  if (stopping criterion is fulfilled)
(3)       Make a leaf node (node N)
(4)  else
(5)       Choose the best attribute and make a test node (node N) that tests the chosen
          attribute
(6)       Divide cases into subsets according to the values of the chosen attribute
(7)       Generate a decision tree for each subset
```

As a result, the key questions are proposed:

- How to select the best attribute? Alternatively, what's the attribute selection criterion?
- When to stop the recursive splitting? Alternatively, what's the stopping criterion?

These questions will be discussed via an algorithm ID3 (Iterative Dichotomiser 3), which is a relatively simple algorithm and robust enough to solve the binary matrix data being used in this thesis [37].

Before mentioning to ID3, the attribute selection criterion should be explained. At first, the criterion requires the attributes adequate. That means all the cases which have the same attribute values belong to the same class. Moreover, if the attributes are adequate, it is always possible to create a decision tree which can classify the training data into right classes as long as the attributes are adequate.

Attribute selection criterion's aim is to generate simple (small) decision trees. Derives from the principle called Occam's razor [37]: If there are two models having the same accuracy on the training data, the smaller one (simpler one) can be seen more general and thus better. It will be more general if the tree is smaller. Moreover, the smaller tree is easier to understand and possibly more correct in classifying unseen cases. Therefore, generating simple nodes are one of the measures.

The complexity of a node is defined as its largest when the node has an equal number of cases from every class occurring in the node. Moreover, it will be the smallest when the node has cases from one class only. Heuristic attribute selection measures (measures of goodness of split) are used. These aims to generate homogeneous (pure) child nodes (subsets).

Then comes to ID3. It is an early decision tree algorithm having following assumptions:

- Attributes are categorical (qualitative) and have a small number of possible values.
- The class (the target attribute) has two possible values. However, the algorithm can be easily expanded for classification tasks with more than two classes.
- Attributes are adequate.
- Data contain no missing values.

ID3 selects the best attribute according to a criterion called information gain. Criterion selects an attribute that maximises information gain (or minimises entropy).

After the attribute selection criterion, the ID3's stopping criterion is indicated:

- ID3 assumes that attributes are adequate.
- It splits the data in a recursive fashion until all the cases of a node belong to the same class.
- The class of a leaf node is defined by the class of the cases in the node. If the leaf is empty (there are no cases with some particular value of an attribute), the class is unknown (the leaf is labelled as 'null')

As a result, a decision tree is efficient. However, ID3 is good at classifying data into two classes. To make decision tree suitable for the data that the thesis will use, the C4.5 algorithm is helpful. All in all, k-nearest neighbour classifier and C4.5 decision tree will be compared via being applied to realistic data. Then which algorithm to use will be decided.

3.4. Validation and tools

Cross-validation, as its name proposes, divided learning data into some parts and use one part as test data, and other parts as training data. Then do the validation with these data.

To be specific, the initial data are randomly divided into k subsets or folds; the subsets need to be mutually exclusive. S_1, S_2…S_k, Each subset is the similarity of the equal size. k-1 subsets are used as the training data, and the k^{th} subset is the test set. First the subsets S_2… S_k form the training data and the subset S_1 the test set. Then the subsets S_1,

$S_3 \ldots, S_k$ can form the training set and the subset S2 as the test set. Moreover, loop the process until each subset is employed once as a test set [36].

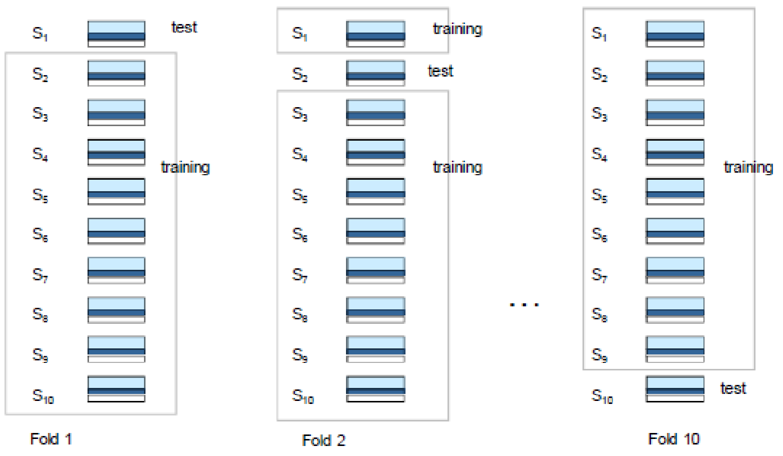


Figure 12 demonstration of 10-fold cross-validation [36]

Typically, the value of k is 10. Therefore, the process is called 10-fold cross-validation. A performance measure such as the accuracy of a classification can be calculated in this validation, being as the mean of the measures obtained in each iteration (the average of the measures from the k test sets). Moreover, the performance measure in this validation can also be based on the number of correct predictions/classifications from the k iterations.

In classification tasks, a stratified sampling is normally used. Moreover, the distribution of the class (the target attribute) in each subset is almost the same as that in the initial data. Cross-validation can be applied any times via different partitions of the initial data. Therefore, a performance measure is calculated as the mean of the measures acquired in different cross-validation trials.

The actual classifier is generated from the initial data (the whole data). The accuracy of the classifier on future cases is estimated with k-fold cross-validation. In general, stratified 10-fold cross-validation is recommended for estimating accuracy. This recommendation is based on both theoretical and empirical results [35].

There is a special case of cross-validation, and it is called leave-one-out. That means, if the data has n cases, leave-one-out (n-fold cross-validation) is done in the following matter: n-1 cases form the training data, and the nth case form the test data and so on. Then each case is employed once as a test set. Therefore, leave-one-out can be done only once. Furthermore, the method is employed with small data sets (hundred cases or less).

Applying this validation method to a k-nearest neighbour, the accuracy of the prediction/recognition will be calculated. In addition, according to the result of accuracy, the best k for k-nearest neighbour can be computed by the following process with pseudocode:

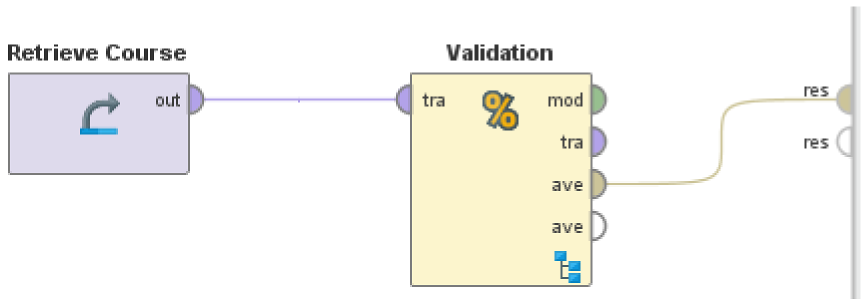
```
Divided original data into m fold (m-fold cross-validation)
loop (start with k = 1)
    loop (start with the first subset of test data)
        Apply the k-nearest neighbour process to training data (other subsets)
        Compare the prediction results to test data’s classes
        Calculate the accuracy via the test data
    until each subset is used
    Compute and save the accuracy of the total cross-validation with k
    Increase k by 1
until k = 10
Compare all accuracies and select the value k with the highest accuracy
```

However, design validation algorithm to every testing learning algorithm is inefficient since it will take more time on coding. Therefore, some data mining tools such as Weka and Rapidminer can be utilized as the validation tool. As data mining tools are just used for validation, the tools will be introduced briefly.

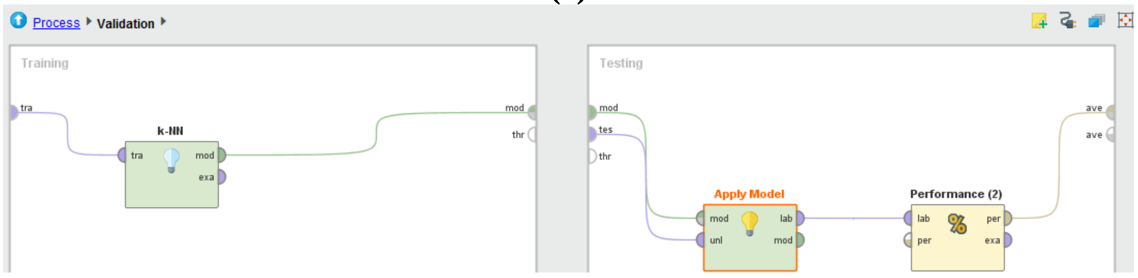
Weka, refers to Waikato environment for knowledge analysis, is an open source data mining software in Java, and it is still being developed [43]. With this tool, the common data mining and machine learning algorithms can be applied to dataset directly, including classification, regression, clustering, association rules and validation. Furthermore, Weka is also available for big data. For instance, apply the valid dataset to the ID3 decision tree and 10-fold cross-validation to Weka, it will do the classification and validation at a time. Then the validation result (accuracy) will be published on the software’s interface.

Valid dataset means the structure of a dataset is suitable for the learning algorithm, for example, the dataset without a class label is invalid for supervised algorithms like k-NN and decision tree. In the example above, only the dataset with a class label can be applied to ID3 decision tree, if the dataset has no class label, Weka will not work and show the warnings. Furthermore, some parameters can be set by the user. For instance, the k of k-NN can be set; the result will also be calculated with the value of k. The number of the folder in x-fold cross-validation can also be set, the user can choose the reasonable value of x as his wish, and the leave-one-out is also an available option.

Rapidminer is the similar application to Weka. Thus it can also deal with the machine learning problem via various algorithms such as validation, classification, and other solutions. Rapidminer does well in the visualisation of modelling; users can design and draw their model of machine learning as it indicates in Figure 130



(a)



(b)

Figure. 13 Example designed model of cross-validation in Rapidminer

In Figure 13 (a), retrieve course block is the inputting data; there is a line connecting the data to validation module. Then the validation module connects its results to the out of the screen. Moreover, the validation module can be set concretely (Figure 13 (b)). First is the training part, the algorithm to be validated should be put here and connected to the input data and testing part. In testing part, there is an applied model for processing the raw data. The performance block is for choosing the performance and structure of the results.

As a result, Rapidminer is a useful tool for helping users understand the process of machine learning more clearly. Furthermore, if users click “run” button, the process will work and form the result list to the screen. In Figure 13, the example is using validation module. Thus, the result list will be the accuracy of the learning algorithm when to apply data to it.

4. CAPTCHA Recognition Process

The CAPTCHA recognition process will be presented as the robustness test. To test the CAPTCHAs' robustness, the process of cracking CAPTCHA should be explained clearly. Thus, the elements that are concerned with robustness can be derived via analysing the details of the process. The element that makes the cracking cost more time or makes the accuracy lower can improve the robustness of CAPTCHA. However, as this thesis will focus on machine learning part, the elements related to digital image processing will be ignored.

Before the test, some hypotheses should be set. As the thesis will focus on machine learning and the digital image processing will be despised, the processed image data would be considered as correct enough (over 90% noises have been removed, and characters are segmented completely). Therefore, the cleaning data step of machine learning will be ignored. However, the validity of the data will be proved via the validation of robustness test at last since invalid data cannot keep a high recognition accuracy.

The method for robustness test is as follows: Firstly, analyse the chosen CAPTCHA, and to identify the parameters that can impact robustness. Secondly, pre-process (de-noise, segmentation…) the learning data (CAPTCHA image). Thirdly, select reasonable learning method. Next, design learning algorithm according to the method chosen in the last step, analyse the process and find out elements that can disturb machine learning. Finally, validate the algorithm via cross-validation to make sure the algorithm is correct and efficient. The expected success rate of the test would be 90% as long as the pre-processed part can process images well.

The outcome of the test will be discussed Chapter 5. In Chapter 2, some metrics standard of robustness are also mentioned, among this accuracy will play an essential role in the robustness test. At first, accuracy is intuitive evaluation standard for the robustness test. In a right process, accuracy could be considered as the success rate. Secondly, accuracy can help to select the machine learning algorithms. Though processing time can also help with that, accuracy is more intuitive and easy to calculate. At third, comparing with precision, processing time of machine learning part impacts robustness less since the basic learning algorithms more or less have similar time complexities. Furthermore, in one recognition process, there are only 4 to 12 characters (depending on the string length of CAPTCHA) to be recognized. Thus, the processing time is usually below three seconds. As a result, the selections in the recognition process will mainly rely on accuracy.

The testing environment is here:

- Operating system: Windows 10 (64-bit)
- Central process unit: Intel(R) Core(TM) i7-4700MQ CPU @ 2.40GHz
- Random-Access Memory: 8.00 GB
- Programming language: Python 2.7
- Method: Custom method based on OCR
- Validation: 10-fold cross-validation
- Validation tools: Weka and Rapidminer

4.1. Pre-processing Target CAPTCHA Analysis

As machine learning needs quite many data, the data must be free and easy to get. CAPTCHA in captchas.net is free to use, and it applied API (Application Programming Interface) in six network programming language. Furthermore, captchas.net has a backup server in another network guaranteeing; this provided stable services for the websites. Since 2004, there has been no downtime of both servers at the same time [1]. As a result, this kind of CAPTCHA is widely used, and the testing data can be generated easily and efficiently.

Properties of CAPTCHA in captchas.net:

- Alphabet/Character set: [a,b,c,d,e,f,g,h,k,m,n,o,p,q,r,s,t,u,v,w,x,y,z]
- Number of letters: 6
- Width, Height: [240, 80] pixels
- Colour: Two colours, white and another colour.
- Random or dictionary: Random

There are three distortion methods which are applied for CAPTCHA generation, and they are noise pixels, rotation of characters and translation of characters. As shown in Figure 14, there are so many noise pixels that they can be even considered as the background. The noise pixels which are inside the characters turn the pixels of characters to white. The rotation of characters is slight since the angles of rotation will never reach over 30 degrees. It is clear that the characters are located in different horizontal axis, which is implemented using the translation methods. These three kinds of distortion will not impact usability significantly because the distortion level is low and there are no confusing characters included in the character set. Furthermore, the translation is mild and only forward to up and down.

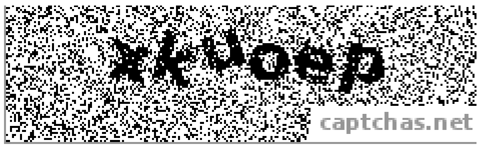


Figure. 14 sample CAPTCHA image from captchas.net [1]

The alphabet set excludes the letter “i”, “j”, “l”, which improves the usability of CAPTCHA due to the reduction of confusing words. Though there is no number in the character set, users never know the truth that no number in the CAPTCHA at all, The exclusion of confusing letters such as “i” , “j”, “l” helps users from confusion with the number “1”. In addition, just as it showed in Figure 14, there is an “o” in the CAPTCHA, it is similar to number “0”. However, as other five characters are clearly letters, the user will feel it more like “o” instead of “0” and this makes CAPTCHA more predictable. If there is another similar character such as letter “l”, the user may treat it as number “1”, then the appearance of number “1” changes the possibility of “CAPTCHA may have number” to over 0%. Users may think “there may be some numbers” instead of “They are all letters”. As a result, the user may type “1” and “0” while the characters are “l” and “o” in fact.

The number of characters is six which is a common number for CAPTCHA. As the CAPTCHA from captchas.net is using random string and a large number of characters is helpless for users to predict, it is better to have less number of characters. As a result, the number from four to six are the reasonable numbers of characters.

As for the presentation of this kind of CAPTCHA, the font type is the typically print hand words, and the characters are bold. This increase the usability since print hand words are the standard form of letters, and bold characters have more attraction to users. Therefore, users can recognize the characters in an average standard time. Comparing with the whole image of this CAPTCHA, characters occupy around 12.5% to 17.5% space. It will be clearer if the size of characters is a little larger. However, as the characters are already easy for users to recognize, the larger size of characters make little impact on usability.

The image size is changeable, the typical width and height are [240 px, 80 px] (px refers to the pixel). This CAPTCHA uses two colours, white and another colour. These two colours are used for different part of CAPTCHA: white is for the background and the noises inside characters; another colour is for the characters and most noises. In Figure 14 another colour is black. Moreover, the colour for characters and noises can be set as another colour. As the colours are only two and easy to distinguish, the use of colour in this CAPTCHA cannot increase the robustness. Nevertheless, the settable use of colour can enhance the usability if the colour of characters matches the design of webpages' interface.

4.2. Data grabbing

As captchas.net provides efficient API for Python, grabbing the data (CAPTCHA images) is easy to implement. Tools used for data capture were captchas.net web service, Python scripts, Windows cmd file, and wget. The data for CAPTCHA images was captured using the services provided by CAPTCHAS.net [1]. It is a free service that provides captchas - even for commercial use. The site has a form to generate single captchas and also publishes ready-to-use examples for the following languages: PHP, ASP, Perl, Python, JSP, and Ruby. From these options, we opted to use Python scripts. Original scripts were downloaded from the site (CaptchasDotNet.py and CaptchaGen.py). CaptchaGen.py generates an HTML page containing a form that's used to request a captcha image from the service. CaptchasDotNet.py has the logic to generate a random string, encrypts the random string, and generates a URL for CaptchaGen.py to request the captcha image. CaptchasDotNet.py was modified to emit the (plain text) captcha text and the generated URL. The information about the captcha was embedded in a command line to call to wget application. A single call of CaptchaGen.py will now generate the following command line:

```
wget -O anovhs.bmp
```

It is necessary to mention that CAPTCHA text has been encrypted as the random parameter, and the parameter –O [output direction] for wget exposes the captcha text as the filename. With this implementation, calls to CaptchaGen.py are repeated for each captcha image – text pair. Naturally, this is slow but can be easily automated using a windows command line command. To generate 100 captchas we used:

```
:\> for /L %f in ( 1, 1, 100 ) do CaptchaGen.py
```

This generated a command file captchacrawl.cmd contains wget commands for the requested CAPTCHAs. GNU Wget is a free software package for retrieving files using HTTP, HTTPS and FTP, the most widely-used Internet protocols [44]. It is a versatile tool that can be very useful for many kinds of data gathering and mining tasks.
To summarize the workflow:

1. Issue requested number of calls to CaptchaGen.py using the for - command
2. Run captchacrawl.cmd

As a result, in the above example case, we get 100 image files with captcha text as the file name. The generation of the 100 files takes about 2 minutes (including generating the command line and running it). Execution time depends mostly on the response time of the captchas.net server, so it is scales nicely even to bigger amounts.

In the author's grabbing process, the data can be grabbed completely and effectively. The CAPTCHA images crawled from captchas.net are with the same distortion methods and character set at any time. Therefore, the grabbed CAPTCHA images can contain most possibilities of distortion methods and characters, which makes the recognition process much easier.

4.3. Processed CAPTCHA

As Figure 15 indicates, the digital image processing part is conducted. The Input CAPTCHA files are de-noised and smoothed [45] via image-processing algorithms. De-noising and smoothing are both based on filter operations. One of the commonly used de-noising [40] algorithms are flood-fill, but they will not work well for the approach because they are designed to find fewer, larger sections, rather than lots of small sections. An open source image processing [46] library ImageJ [50] was used in the process. Collections of various digital image-processing filters, such as mean filter and median filter, were applied to meet the desired results.

On the segmentation [40] stage, we needed to slice the CAPTCHA image into multiple characters segment images. An Optical Character Recognition (OCR) library was used to split our CAPTCHA image. Each crawled CAPTCHA after being de-noised was segmented into multiple processed bitmap images.

Segmentation [46] creates the different segmented images. In binary bit stream stage these segments are transformed into 16 x 16 binary matrices [45]. Where in the binary matrix one value represents black pixel value and 0 represents white. These values will make the instance of the dataset, which will have a character class. A bitmap segment to binary matrix transformation algorithm was written for this purpose.

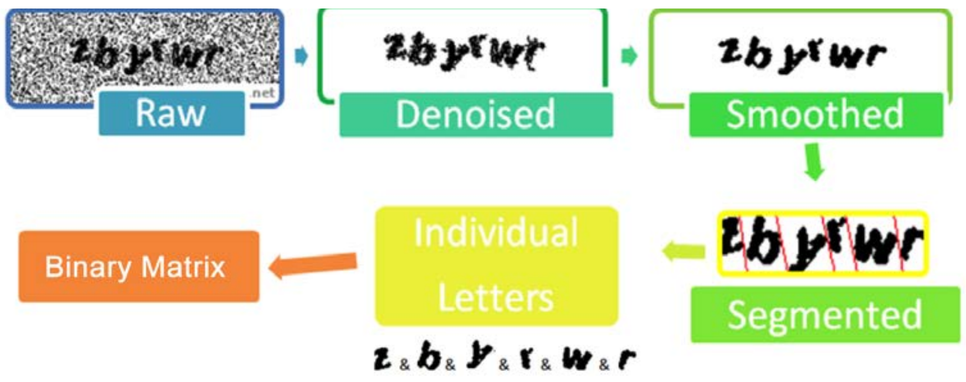
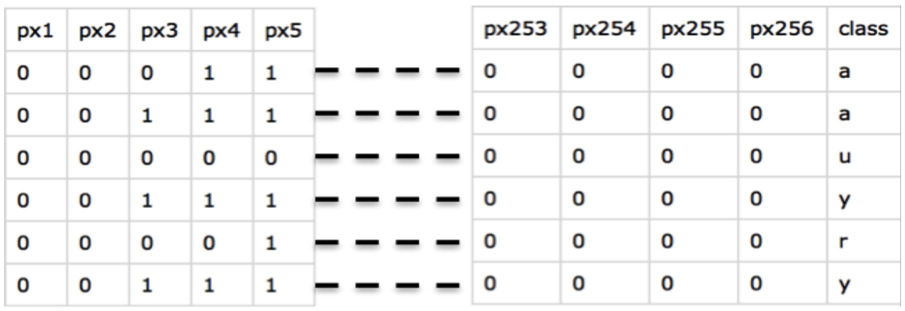


Figure. 15 process of pre-processing

Processed data was organized in folders of their respective CAPTCHA's. Each folder contains de-noised bitmap CAPTCHA, segmented bitmap files with the character as their filename and a CSV file containing binary matrices of each segment with its character class. Data from crawled CAPTCHA images gave us 27865 binary matrices (27865 cases and one label line).



px1	px2	px3	px4	px5		px253	px254	px255	px256	class
0	0	0	1	1	— — — —	0	0	0	0	a
0	0	1	1	1	— — — —	0	0	0	0	a
0	0	0	0	0	— — — —	0	0	0	0	u
0	0	1	1	1	— — — —	0	0	0	0	y
0	0	0	0	1	— — — —	0	0	0	0	r
0	0	1	1	1	— — — —	0	0	0	0	y

Figure 16 CSV structure of processed a CAPTCHA image

Data for building a model were obtained in the pre-processing phase. Data contain 27,865 cases with 256 attributes. In Figure 16 the attributes present a binary matrix corresponding to a single character read from a CAPTCHA image. The attributes are coded by its pixel identifiers: [px1, px2, px3, px4…… px254, px255, px256, class]

For example, one case is presented as a binary matrix in the following form:

[0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,
0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,1,0,1,
1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,a].

There are 23 class values corresponding to Latin alphabet letters: a, b, c, d, e, f, g, h, k, m, n, o, p, q, r, s, t, u, v, w, x, y, z. They are the same as the character set that is mentioned in Section 4.1. Therefore, there are no missing characters from the dataset.

4.4. Preparation of Character Recognition and Prediction

First of all, the classification algorithm should be decided. As it mentioned in Section 3.3, the candidate algorithms are k-nearest neighbour and C4.5 decision tree. The detail validation results (Class-wise accuracy) is given in Appendix 1.

As a data mining tool, Weka/Rapidminer and its J48 algorithm which is C4.5 decision tree algorithm was chosen. The resulting decision tree consists of 251 leaves and has the size of 501. 10-fold cross-validation was engaged for validating the model. As the result, 27,721 cases (99.48%) were classified correctly, and 144 cases (0.52%) were classified incorrectly. Thus, 10-fold cross-validation approved high accuracy of the constructed model.

Weka’s IB1 algorithm was applied to the same dataset to check the classification with the k-nearest neighbour method. 10-fold cross-validation was engaged for validating the model. As the result, 27,707 cases (99.43%) were classified correctly, and 158 cases (0.57%) were classified incorrectly. However, the divergence of accuracy in the cross-validation is still small. Therefore we can state that k-nearest neighbour classified the data fairly well.

Though there are two candidates, one more algorithm could be tested as a comparison. It is called Naive Bayesian classifier. Naive Bayesian classifier is also a supervised learning algorithm. However, when it applied to Weka/Rapidminer and the dataset, 24,773 cases (88.90%) were classified correctly, and 3092 cases (11.10%) were classified incorrectly. This classifier recognized CAPTCHA symbols worse than two previous classifiers. As a result, there is no need to introduce this classifier anymore.

As the results indicate, k-NN and C4.5 decision tree both can perform well in the CAPTCHA recognition with such a large dataset. To make the difference clear, another test is conducted. The contents of the dataset will be reduced to only 353 cases. Moreover, the cases even include the characters “i” and “l” which are so similar that can make recognition easy to make mistakes. The purpose is to judge which algorithm can perform better in such a restrict condition. The test result is shown in Table 2. The first column is the result of the test under the normal condition, and the second column shows the result of the restricted test.

	Accuracy (27865 cases)	Accuracy (353 cases)
K-nearest neighbour	99.43%	81%
C4.5 decision tree	99.48%	45%

Table 2 Test with restrictions

As shown in Table 2, K-nearest neighbour's accuracy drops slightly (99.43% to 81%), the accuracy C4.5 decision tree decreases dramatically (99.48% to 45%). As a result, K-nearest neighbour performs much better than C4.5 decision tree. It proves that k-nearest neighbour algorithm is more suitable for the processed CAPTCHA data. Then k-NN is chosen as the CAPTCHA robustness test kernel algorithm in machine learning part. Furthermore, some confusing characters can improve the robustness of CAPTCHA, and a limited number of data can also enhance the robustness.

4.5. K-Nearest Neighbour

There are many distance measures for binary data [34]: Euclidean Distance, Manhattan Distance, Jaccard Similarity, Dice Similarity and other numerical measures as Section 3.3.2 indicates. With the pre-processed result of CAPTCHA, all images are transferred into binary data, which means the values of attributes are 0 and 1. In this situation, Euclidean Distance has no difference with Manhattan Distance. In addition, 0 means the background of CAPTCHAs, so most values are 0, and 1 is the value matters. Therefore as for the similarity measure, Dice Similarity, which provides more weight to value "1", is better.

As for the value of K, it is mentioned in Section 3.3.2 that the time complexity of classifying one case via k-nearest neighbour is $O(k \cdot n)$ for n is the number of lines of training data cases, and k is the number of the nearest neighbour. As a result, k can impact the cost of recognition. In Table 3, the accuracies of k-nearest neighbour classification with a different value of k are listed. The accuracies decrease quite slightly while the value of k increases. Therefore, for the processed CAPTCHA data, the best value of k is one (the nearest neighbour).

	K = 1	2	3	4	5
Accuracy	99.51%	99.49%	99.49%	99.47%	99.47%

Table 3 Accuracies with different value of K, formed by the method given in Section 3.4

The main process by using k-NN is given here. Firstly, take the data to a dictionary or a list for storing, which means the learning in k-NN. Moreover, I made a function compute the distance/similarity between unpredicted data and learning data. Secondly, add all the result of distance/similarity function to a list and sort them, while only keep the indexes. Thirdly, the list has an original order while another list has the list of indexes which are sorted according to the results of distance/similarity function. At last, choose the maximum one if using similarity measure or minimum one if using distance measure as the answer for prediction.

As for the optimization of time complexity, the k-dimensional tree method (KD-tree) is one of the methods. It is a data structure that can reduce the cost of finding the nearest neighbour to O(log n), n is still the number of cases, and the time complexity of construction of a tree is $O(n \cdot \log^2 n)$ via naive algorithm. It seems that construction of a tree may cost more time. However, one construction for the data is efficacious forever. Thus, it can reduce the processing time for series of recognition.

Nevertheless, KD-trees will be weak if the data has too many dimensions which require the algorithm to visit much more branches than in low-dimensions spaces. As a result, KD-trees cannot help the recognition of CAPTCHA due to each case has 256 attributes (or each case is the point in 256 dimensions).

As a conclusion, the selected distance measure is Dice Similarity and Manhattan Distance. In addition, the best value of K is one. Thus, the value of K used in the process is one as well.

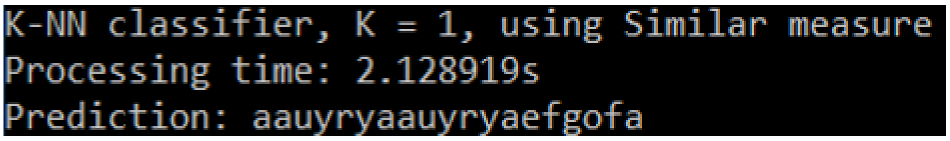
4.6. Test algorithm

The pseudocode of machine learning part of testing algorithm is here (words in bold are command and parameters):

```
Receive the parameters, k, [distance/similarity] from console
Set result_list as “” (empty string)
Start counting time
for each case x in target data
    for each case n in training data
        Compute distance/similarity between x’s attributes and n’s attributes
        if the distance/similarity is below/over last distance/similarity (first
        distance/similarity is 100/0)
        then set a as n’s class
    end for
    Store the value of a to result_list
end for
end counting time
```

As the k is determined as 1, majority voting is unnecessary here and just find out the nearest neighbour which can be done in the same layer of a loop. Then the **result_list** contains all the recognition results and **time** is the processing time. Source codes of this program will be written in Appendix 2.

The result is shown in Figure 17 indicates. 19 cases of processed CAPTCHA images are used as the test data. The processing time indicates the time of classifying these 19 cases into their classes. To be concrete, the time includes the time of storing training data, computing the similarity and selecting the nearest neighbour. Prediction is to show the results of recognition.



```
K-NN classifier, K = 1, using Similar measure
Processing time: 2.128919s
Prediction: aauyryaauyryaefgofa
```

Figure. 17 Example result of test program

As the accuracy was validated in Section 4.2, the test result is valid and correct, and the automatic recognition of CAPTCHA is realised.

5. Results and Discussions

Factors of affecting the usability of CAPTCHA have been presented in Chapter 2. They help to analyse the selected CAPTCHA in Chapter 4. Furthermore, as some measures for improving robustness have an adverse impact on the usability of a CAPTCHA, the factors are further discussed and adjusted in this chapter for the improvement of the robustness of a CAPTCHA.

Since this thesis is focusing on the machine learning part instead of digital image processing part, the results which are concerned with image processing will not be discussed in this Chapter. Only some elements which influence the machine learning process (e.g. grabbing data process) are discussed here.

In Figure 18, it indicates the process of CAPTCHA image's automatic recognition. The rectangles are the statues and processed results. The circles mean the "action" steps. The first rectangle is the CAPTCHA generation which means the original raw CAPTCHA images on a website. The raw CAPTCHA images are crawled by the crawling step and stored to the local devices. The stored CAPTCHAs are then pre-processed by digital image processing and transformed to processed data which is in a digital matrix with little noises. The processed data can be used for the recognition process via machine learning, and finally, the recognition results will be given in the prediction. As a result, the discussions in this chapter are based on the details of the three circles (crawling, digital image processing and machine learning).

In Chapter 4, the particular process of automatic CAPTCHA recognition is described and realised successfully. The process is shown in Figure 18. Then some elements that can impact the recognition process can be discovered through the process. To be specific, if an element, such as diversity of data, can force the recognition to cost more time or can decrease the accuracy, it will be the positive element for improving robustness. For instance, a factor of data forces the program to store more data to keep an ideal accuracy. This factor is the positive element for improving robustness.

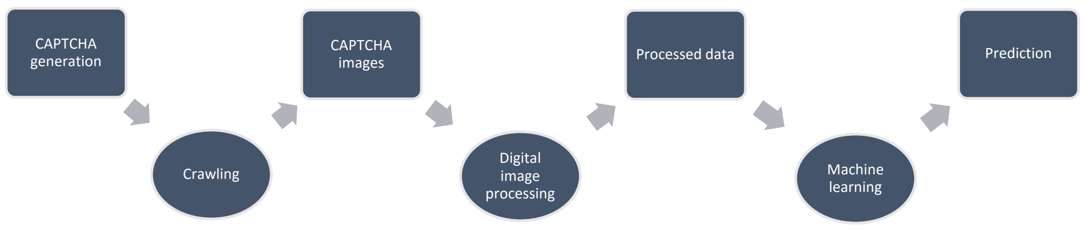


Figure. 18 Process of CAPTCHA's automatic recognition

5.1. Confusing characters

Some confusing characters can improve the robustness of CAPTCHA. In the restricted condition test for selecting learning algorithms, a dramatic decrease of C4.5 decision tree's accuracy can be found (From 99.48% to 45%). The accuracy of K-NN performs better than C4.5 decision tree. Nevertheless, it still dropped from 99.43% to 81%. Therefore, the confusing words can obviously improve the robustness of CAPTCHA via decreasing the accuracy.

Furthermore, in Table 4, the individual accuracies of "i" and "l" are listed, and the values of them are obviously much less than the average precision. Especially the accuracy of "i" even reaches the 0.00% which means no correct recognition is done by this approach. For the perspective of K-NN, the confusing characters create a huge barrier for the recognition. The accuracy of letter "i" is 25%, and accuracy of letter "l" is 12.50%. Both of them are quite lower than the average accuracy (81.33%). It proved that the confusing characters could improve the robustness of a CAPTCHA.

	Accuracy of "i"	Accuracy of "l"	Average Accuracy
Decision tree	0.00%	12.50%	45.84%
K-NN	25.00%	12.50%	81.33%

Table. 4 the accuracies of recognizing "i" and "j" in the restricted condition test

Nevertheless, as it mentioned in Section 2.4, confusing words are harmful to the usability of CAPTCHA. In such a noise environment, users can also make wrong recognitions. Therefore, this factor is ineffective to improve the robustness while keeping the usability.

5.2. Diversity of data

Increasing of the diversity of each class of CAPTCHA data enhances CAPTCHA's robustness via force automatic recognition cost more time. There are two approaches to increase the robustness of CAPTCHA, i.e. significantly increasing the number of learning data which will increase the cost time of machine learning process and the difficulty in reaching an acceptable accuracy. Moreover, it may cause the over-training (or overfitting). In a machine learning process, the more possibilities of the recognition target, the number of data it requires. The increase of diversity of each class of CAPTCHA data means there are more than one possibilities of the data from a class. For instance, [1,1,1,0] belongs to class "a", [1,1,1,1] belongs to "b", then the training just needs two cases to do recognition. If [1,0,1,1], [1,1,0,1] also belong to class "a", [1,0,1,0] belongs to class "b", the training may need more cases to do the training.

Overtraining (overfitting) is a common problem in many classification algorithms; it means that in the training dataset, more cases belong to a class than another class while the situation is opposite in the real test dataset. For instance, in the training dataset, ten same cases [1, 1] belong to class “a”, only one case [1, 1] belongs to class “b”, then if the test data case is [1, 1], the prediction will be class “a”. However, in real test dataset, there are 99 cases [1, 1] more belong to class “b”, and class “a” still has those ten cases. Therefore, the prediction will make a mistake when judging the ownership of case [1, 1]. The main reason that causes overtraining is that some noises still exist in the training dataset, or the training dataset is insufficient. The increase of diversity of each class of CAPTCHA data can cause the lack of training data since it increases the total amount of data. In addition, some specific measures to realise this factor can make some noises to the dataset. However, as the digital image processing is not the emphasis, the noises will be regarded as being processed perfectly as long as the data is valid. In Chapter 4, it has been shown that the more data being applied to k-NN, the more time the recognition will cost. As a result, the increase of diversity of each class of CAPTCHA data will enhance the robustness.

As the time complexities of the classification and validation, which are mentioned in Section 3.3.2 and 3.4 and tested in Chapter 4, indicates, it costs too much time on classifying and validating accuracy with a large quantity of data. Thus, another custom program just takes 23 * 50 = 1150 cases out of the original data, and each letter has 50 cases for learning, which might be enough for keeping a high accuracy. Then take them to a dictionary or a list for storing, which means the learning in k-NN. Moreover, a function was made to compute the distance/similarity between unpredicted data and learning data. Then add all the result of distance/similarity function to a list and sort them. However, only keep the indexes. The list has an original order while another list has the list of indexes which are sorted according to the results of distance/similarity function. At last, choose the maximum one if using similarity measure or minimum one if using distance measure as the answer for prediction.

Nevertheless, taking 1150 cases is harmful to the accuracy of recognition since the test results are negative (the accuracy decreases dramatically to 60%). The test is using the 1150 cases to do the training, and the origin data is for the test data. As a result, 50 cases for each character cannot explore all the possibilities of the character’s data. In another word, the increase of diversity of characters’ data can help improve CAPTCHA’s robustness.

Specifically, in k-nearest neighbour, a larger diversity of processed CAPTCHA data can cost more time via forcing attackers to choose larger k. In Section 4.4, the time complexity of K-NN is found as O(k • n), thus the larger k is the more time will cost. In the robustness test in the thesis, as the diversity of training data is insufficient, one is enough for the best value of k. However, if there are more than one class having similar attributes, the value of k need to be increased since there may be more than one case which has the same similarity. In this situation, a majority voting is necessary. If the value of k is one, the prediction may be wrong, for example, for the cases which have the same similarity/distance, the first case's class is "a". Therefore, the prediction will be "a". However, the following cases belong to "b". That means there are more "supporter" to "b" and "b" should be the right answer in common.

5.3. Measures to enhance the diversity of data

The diversity of CAPTCHA data is determined by the distortion, contents of CAPTCHA as well as image processing process. One purpose of image processing process is to decrease the diversity of data after all. However, there are so many methods to improve the diversity of CAPTCHA data that cannot be listed all here. Furthermore, most methods are concerned with digital image processing and not every method is effective to the diversity of CAPTCHA. Therefore, as this thesis is using CAPTCHA from captchas.net as a case study, the methods which are used in this kind of CAPTCHA will be discussed.

Adding noise pixels is one of the distortion methods in CAPTCHA from captchas.net. However, it has been de-noised by the custom algorithm and has little impact on the diversity of data after being de-noised well. Moreover, de-noising is the necessary process for every CAPTCHA recognition algorithm. Therefore, this method is creating trouble for digital image processing part. If there are still many noises in the processed data, the machine learning part cannot even start. Therefore, it is more concerned with digital image processing, which is not this thesis focusing on. Thus this method will not be discussed here.

Another distortion method of this CAPTCHA is a rotation. In Chapter 4, the rotated characters are kept without any process, and only the letter "o" is full symmetry. Therefore, a slight rotation can increase one kind of data of a character. To be concrete, a character rotated by 5 degrees clockwise will have different data formulation with the character rotated by 6 degrees. The 1 degree (or even fewer) difference will create a clear difference between the characters data. As it indicated in Section 3.2, the rotation is complemented process in pre-processing of CAPTCHA recognition, and it is unnecessary since rotation will not form noises directly and rotated characters can also be learned by computer. Due to the same reason, the attacker must make a decision that adding rotation process to pre-process or cost more time on machine learning. No matter which choice the attack takes, the processing time, which can quantize robustness, will increases.

Besides the distortion methods, enlarging character set could be a good measure as well, since it can increase the diversity of training data. However, as it was mentioned in Section 2.4, too large character set will decrease the usability of CAPTCHA. To be concrete, the larger character set will have a larger diversity of processed data. However, besides the confusing characters, too many kinds of characters, such as capital letters, numbers, and symbols, will decrease the predictable factor in usability. If the character set has only one type of characters, such as lowercase Latin letters, users can easily know the answer will be in lowercase Latin letters when they see the CAPTCHA. If the number is added to the character set, users may consider more when they see the numbers “0” and “1” which are similar with letters “o” and “l”. That means what you see is not always what you get. Thus it decreases the usability. Nevertheless, there are also some measures to enlarge character set while just decrease usability slightly. Use both capital letters and lowercase letters as the character set while the answers can be accepted in any combination of capital letters and lowercase letters, which means making CAPTCHA case-insensitive. For instance, CAPTCHA shows “AaOoBC”, users can input “aaoobc”, “AAOOBC”, “AaOoBC” and other similar string as their answers and the answers will be counted as correct. In this situation, the character set doubles the training data than the data from the character set using only lowercase letters or capital letters. Furthermore, a hint of “The answer is case-insensitive” will make the decrease of usability slighter. As a result, this factor will work in the specific situation.

5.4. Crawling effectiveness

The last factor that has been found in the process is the difficulty of crawling effective and complete data. It is located in the grabbing data process. In Section 4.1, enough data can be collected in a short time, and the data that has been collected is valid. Moreover, most possibilities of distortion of CAPTCHA characters can be collected as long as enough data has been grabbed. Therefore, one measure can be derived. It is to create barriers for providing all information of CAPTCHA’s distortion and content. To be concrete, provide only a few kinds of distortion or content of CAPTCHA at a time.

For instance, CAPTCHA generation program can use an encrypted algorithm that generates a kind of CAPTCHA characters according to a parameter such as hours or the date. Then the difficulty of crawling complete data will be increased dramatically since the data crawled in one day cannot cover all the possibilities of CAPTCHA characters. Thus, the data on a specific day cannot be used for recognition of another day’s

CAPTCHA if the parameter is a date. Furthermore, it will take much more time to grab enough data. This method is quite effective to improve the robustness of CAPTCHA unless the encrypted algorithm is cracked. What is more important is this method will never impact usability since the method is about the CAPTCHA generation program (backend) where users cannot reach. For instance, if apply this method to the CAPTCHA from captchas.net, the method will refer to the process like this: on Monday, the rotation angles for the characters “a”, “h”, “m”, “x” are 5, 7, 9, 11 degrees in clockwise direction, other characters will be rotated 5 degrees in opposite direction; on Tuesday, the characters “b”, “k”, “o”, “y” will be rotated 5, 7, 9, 11 in anticlockwise direction, other characters will be rotated 5 degrees in opposite direction; and like this other day will have another rotation way. Then it will be trouble when the cracker tries to grab the CAPTCHA’s data.

It is also feasible to change the contents algorithm instead of distortion algorithm. For example, on Monday, CAPTCHA uses the characters from “a” to “m”; on Tuesday, CAPTCHA uses characters “f” to “x”; on Wednesday, CAPTCHA add some number characters; on Thursday, CAPTCHA add some CAPITAL letters and so on. This measure is useful, especially for the CAPTCHA whose character set is large, such as Chinese characters. However, to keep an available usability, the too complicated character set is unacceptable. Therefore, it is enough for normal CAPTCHA that using Latin letters in both capital and lowercase.

As a result, for the users who only see CAPTCHA a few times, creating barriers for providing all information of CAPTCHA’s distortion and content will not change distortion, contents and presentation of CAPTCHA. Thus, the method will not decrease the usability. However, for the crackers who must collect enough data which can cover most possibilities, this method creates a big trouble for them.

Another measure coming from crawling effectiveness is to build barriers to avoid getting CAPTCHA images too quickly in a short time from the same IP address. However, this measure has already been used by many websites and network applications since it is the basic security measure to avoid DoS attacks and spamming attacks. It is similar to the first measure, having little impact on usability as it effects on the backend. Nevertheless, there is an exception: if a user wants to change the CAPTCHA image for some reasons, such as CAPTCHA includes some confusing characters, the user will refresh the web page or the CAPTCHA image for several times. If the user refresh too fast, the user may be considered as an attacker by mistake and his data applied to the website will be denied for a while (usually from 15 minutes to an hour).

As a result, there are three factors which improve the robustness of CAPTCHA derived from the CAPTCHA recognition process: confusing characters, diversity of processed CAPTCHA data and difficulty of crawling effective and complete. Confusing characters is an ineffective factor since it will decrease the usability of CAPTCHA. The diversity of processed CAPTCHA can improve the robustness of CAPTCHA via forcing machine learning part to cost more time on pre-processing. Two measures can increase the diversity of processed CAPTCHA data: some special distortion, such as rotation; enlarging the character set. There are two measures to enhance the difficulty of crawling effective and complete data. One is from the integrity of data, and another one is from the approach of grabbing.

6. Conclusion

This thesis was motivated by the thoughts when the author tried to make a website. As there are many threats, such as DDoS, spamming attacks, and Brute-force attacks, to web security, the CAPTCHA should be robust enough. However, since CAPTCHA is also oriented to normal users, if the CAPTCHA makes users feel uncomfortable or even annoyed, the popularity of the website/application will be decreased. Therefore, the usability is also concerned with CAPTCHA. Robustness of CAPTCHA is oriented to the attackers' robots, while usability of CAPTCHA is oriented to normal users. As they are oriented to different targets, the effects of robustness and usability are also different. However, robustness and usability are still connected since they are determined by all the elements of CAPTCHA.

As a result, the objective was to find out a trade-off between the robustness and the usability of CAPTCHA at first. Then the specific issues of usability could be found. In the usability part, a contribution is to compare the usability of software with the usability issues of CAPTCHA, so that the issues are clearer in details. However, as the robustness is oriented to attackers' robots, attackers' perspective is needed to make the factors more specific. Therefore, the whole process of automatic CAPTCHA recognition needs to be illustrated and realized.

The text-based CAPTCHA was selected as the thesis's focus. Then the methods to recognize CAPTCHA were found in optical character recognition, which is a common method to recognize images. Therefore, the two main process of OCR were clear, pre-process (digital image processing) and machine learning. Attributing to the author's knowledge and interests, the thesis is focused on machine learning part. Then the basic techniques and algorithms were introduced so that the automatic CAPTCHA recognition can be realized. The implementation of the machine learning part of automatic CAPTCHA recognition based on existed basic algorithm was the main contribution of this thesis.

After realizing the automatic CAPTCHA recognition, there were contributions of this thesis that some factors for helping to improve the robustness of CAPTCHA were derived. They are mainly from the machine learning part, and one factor was from the data collection. Nevertheless, these factors need to be compared with the usability issues so that the objective can be achieved.

Nevertheless, there exist some limitations. At first, the digital image processing was not focused on though this part could provide some factors that can improve the robustness. Comparing with the factors given by machine learning part, the factors from digital image processing will be more intuitive and specific since they are about the distortions. Secondly, the example CAPTCHA cannot cover all the types of distortion and contents.

As a result, in the future research, the factors for improving robustness from the digital image processing will be the focus. In addition, more examples of CAPTCHA can be selected for the recognition so that the elements of the text-based CAPTCHA can be compared and more factors can be derived.

References

[1]. Captchas.net, URL: <www.captchas.net>, access date: October 25, 2015

[2]. Feng Bo-Yuan, Ren Mingwu, Zhang Xu-Yao and Suen Ching Y, Automatic recognition of serial numbers in bank notes. In: *Pattern Recognition 2014,* **47**-8, 2014, 2621-2634.

[3]. Takahiko Kawatani and Handwritten Kanji, Recognition Using Combined Complementary Classifiers in a Cascade Arrangement. In: *Proceedings of the 5th International Conference on Document Analysis and Recognition, Bangalore, India*, 1999, 503-506.

[4]. Paul S. Adler and Terry A. Winograd, *Usability: turning technologies into tools*, Oxford University Press, Inc. 1992.

[5]. Glinz Martin, On Non-Functional Requirements, *Requirements Engineering Conference, 2007. RE'07. 15th IEEE International*, 2007.

[6]. Yeshica Isela Ormeñ, Jose Ignacio Panach, Nelly Condori-Fern´ndez and Óscar Pastor, Towards a proposal to capture usability requirements through guidelines, *Research Challenges in Information Science (RCIS), 2013 IEEE Seventh International*, 2013.

[7]. J. Carroll, M., Human-Computer Interaction: Psychology a Science of Design, *Int. J. Hum.-Comput. Stud.*, **Vol. 46**, No. 4, 1997, 501-522.

[8]. L. von Ahn, M. Blum, and J. Langford, *Telling Humans and Computer Apart Automatically*, Comm. ACM, **Vol. 47**, No. 2, 2004, 56–60

[9]. J. Nielsen, *Usability Engineering*, Morgan Kaufmann, 1993.

[10]. Jayaletchumi T. Sambantha Moorthy, Suhaimi bin Ibrahim and Mohd Naz'ri Mahrin, Identifying usability risk: A survey study, *Software Engineering Conference (MySEC)*, 2014 8th Malaysian.

[11]. Sharma, A.K., A. Kalia, and H. Singh, An Analysis of Optimum Software Quality Factors, *IOSR Journal of Engineering*, 2012. **Vol. 2**(4), 663-669

[12]. Westfall, L., Software Risk Management, in International, *Conference on Software Quality*. 2011: San Diego, California.

[13]. Wieland, A., Wallenburg, C.M., Dealing with supply chain risks: Linking risk management practices and strategies to performance. International Journal of Physical Distribution & Logistics Management, **Vol. 42**(10), 2012.

[14]. Xiaotian Zhuang, Rong Pan and Lizhi Wang, Robustness and reliability consideration in product design optimization under uncertainty, *Industrial Engineering and Engineering Management (IEEM), 2011 IEEE International*, 1325 - 1329.

[15]. S. Chong and A. C. Myers, Decentralized robustness, *Computer Security*

Foundations Workshop, 2006. 19th IEEE, 2006

[16]. Durach, C.F. et al. , Antecedents and dimensions of supply chain robustness: a systematic literature review, *International Journal of Physical Distribution & Logistics Management*, **Vol. 45**(1/2), 2015, 118-137.

[17]. Sawik, T., Selection of resilient supply portfolio under disruption risks, *Omega*, **Vol. 41**(2), 2013, 259-269.

[18]. L von Ahn, M Blum and J Langford. Telling Humans and Computer Apart Automatically, *CACM*, **Vol. 47** (2), 2004.

[19]. Jeff Yan and Ahmad Salah El Ahmad, Usability of CAPTCHAs or usability issues in CAPTCHA design, *SOUPS '08 Proceedings of the 4th symposium on Usable privacy and security,* 2008, 44-52.

[20]. Chien-Ju Ho et al. DevilTyper: a game for CAPTCHA usability evaluation *Computers in Entertainment (CIE) - Theoretical and Practical Computer Applications in Entertainment archive,* **Vol. 9**(1), April 2011 Article No. 3

[21]. K Chellapilla, K Larson, P Simard and M Czerwinski, Designing human friendly human interaction proofs, *ACM CHI'05*, 2005.

[22]. Haichang Gao and Wei Wang et al. The Robustness of Hollow CAPTCHAs, *CCS '13 Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, 1075-1086.

[23]. Ahmad Salah El Ahmad, Jeff Yan and Wai-Yin Ng, CAPTCHA Design: Colour, Usability, and Security, *IEEE Internet Computing*, **Vol. 16**(2) , 2012, 44 – 51

[24]. M. Korakakis, E. Magkos and Ph. Mylonas, Automated CAPTCHA Solving: An Empirical Comparison of Selected Techniques, Semantic and Social Media Adaptation and Personalization (SMAP), *2014 9th International Workshop on*, 44 – 47.

[25]. Xu Ming and Jin Zhong, Recognition and Anti-recognition of CAPTCHA, *Nanjing University of Science and Technology,* 2011

[26]. OCR Introduction. Dataid.com. Retrieved 2016-03-24. < *http://www.dataid.com/aboutocr.htm* >

[27]. G. Salton, A. Wong and C.S Yang, A vector space model for automatic indexing, *Communications of the ACM*, **Vol. 18**, 613-620, November 1975.

[28]. P. Baecher et al., CAPTCHAs: The good, the bad and the ugly, In: *Proceedings of Sicherheit 2010: Sicherheit, Schutz und Zuverlässigkeit, Beiträge der 5. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e,***Vol. (GI)**, 5.-7. October 2010.

[29]. Janaina Pilomia, User Experience in Mobile Application Development: Developer and End-user Perceptions, *University of Tampere,* 2011

[30]. K Chellapilla, K Larson, P Simard and M Czerwinski,Building

Segmentation Based Human-friendly Human Interaction Proofs, *2nd Int'l Workshop on Human Interaction Proofs, Springer-Verlag*, LNCS **3517**, 2005.

[31]. Greg Mori and Jitendra Malik, Recognising Objects in Adversarial Clutter: Breaking a Visual CAPTCHA, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, **Vol.1**, June 2003, 134-141.

[32]. Han J, Kamber M, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006, available as: *http://www-sal.cs.uiuc.edu/~hanj/bk2/*

[33]. Tan P-N, Steinbach M, Kumar V, *Introduction to Data Mining*, Addison-Wesley, 2006, available as: *http://www-users.cs.umn.edu/~kumar/dmbook/*

[34]. Langley P, Simon HA, Applications of machine learning and rule induction, *Communications of the ACM* **Vol.38**:55-64, 1995

[35]. Mitchell TM, *Machine learning*, McGraw-Hill, 1997.

[36]. Wilson DR, Martinez TR, Improved heterogeneous distance functions, *Journal of Artificial Intelligence Research* **Vol.6**: 1-34,1997

[37]. Quinlan JR, Induction of decision trees, *Machine Learning* **Vol.1**: 81-106, 1986

[38]. Quinlan JR, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

[39]. Solomon Chris and Breckon Toby, *Fundamentals of Digital Image Processing*, Wiley, 2010

[40]. Ryan Fortune, Gary Luu, Peter McMahon, Cracking CAPTCHAs: Learning to Read Obscured and Distorted Text in Images, Stanford University, also available as: *http://cs229.stanford.edu/proj2008/FortuneLuuMcMahon-CrackingCAPTCHAs.pdf*, April 02, 2016

[41]. Reiss Eric, *Usable Usability : Simple Steps for Making Stuff Better (1)*, Wiley, June 2012

[42]. Bhasin Shweta, *Web Security Basics*, Course Technology / Cengage Learning, 2002, 109-227

[43]. Weka, URL: < http://www.cs.waikato.ac.nz/ml/weka/>, access date: April 04, 2016

[44]. WGET, URL: <http://www.gnu.org/software/wget/>, access date: April 04, 2016

[45]. Ma J., Badaoui B., and Chamoun E., A Generalized Method to Solve Text-Based CAPTCHAs, *Machine learning in Stanford Computer Science*, 2009

[46]. Wen X, *Machine Learning Lecture: Breaking CAPTCHA's*. also available as: *http://www.cs.unc.edu/~lazebnik/fall09/breaking_captchas.pdf*

[47]. Suliman A. Alsuhibany, Optimising CAPTCHA Generation, *Availability, Reliability and Security (ARES), 2011 Sixth International*, 740-745

[48]. Schahram Dustdar, José Luiz Fiadeiro, Amit P. Sheth, *Business Process Management*, 4th International Conference, BPM 2006, Vienna, Austria, September 5-7, 2006.

[49]. Shi Cheng, Yuhui Shi, Quande Qin, T. O. Ting, Particle swarm optimization based nearest neighbor algorithm on Chinese text categorization, *Swarm Intelligence (SIS), 2013 IEEE*, 164-171

[50]. ImageJ, URL: https://imagej.nih.gov/ij/, access date: April 09, 2016

Appendix 1

Class-wise accuracy

Case	**Accuracy**	**Accuracy in 10-fold cross-validation**
a	0.999	0.999
u	1	1
y	0.999	0.999
r	1	1
e	1	1
f	1	0.999
g	0.999	0.999
o	0.973	0.973
n	1	1
s	0.978	0.978
b	1	1
p	1	1
d	1	1
z	1	0.996
v	1	1
t	1	1
h	1	0.993
q	1	1
x	1	1
c	1	1
k	0.939	0.943
w	1	1
m	1	1
Weighted Avg.	**0.995**	**0.995**

Accuracy of C4.5 decision tree

Case	**Accuracy**	**Accuracy in 10-fold cross-validation**
a	1	0.987
u	1	1
y	1	0.999

r	1	1
e	1	1
f	1	1
g	1	0.999
o	0.973	0.980
n	1	1
s	0.978	0.982
b	1	1
p	1	1
d	1	0.986
z	1	0.999
v	1	1
t	1	1
h	1	0.997
q	1	1
x	1	1
c	1	1
k	0.939	0.945
w	1	1
m	1	1
Weighted Avg.	**0.995**	**0.994**

Accuracy of k-nearest neighbour

Appendix 2

Source code of K-NN

```
import time
import csv
from numpy import *
import sys
start = time.clock() #For computing the processing time
f = open("dataset.csv","rU") #training dataset
tf = file("data1.csv","rU") #test dataset
data = csv.reader(f)
dataforp = {}
line = " "
count = 0
k = int(sys.argv[2])
distance_measure = sys.argv[1].lower()
#Preprocessing the dataset
for i in data:
        if count ==0:
                count = 1
                continue
        line = i
        if line[-1:] in dataforp:
                dataforp[str(line[-1:])].append(line[:-1])
        else :
                dataforp[str(line[-1:])] = [line[:-1]]

#Calculate the Manhattan Distance between test attributes and learning attributes
def distance(t1,t2):
        d = 0
        for i in range(len(t1)):
                d += abs(int(t1[i])-int(t2[i]))
        return d
#Calculate the similarity between test attributes and learning attributes
def similar(t1,t2,n):
        sim = [0.0,0.0,0.0]
        for i in range(len(t1)):
                if t1[i] == '1' and t2[i] == '1':
                        sim[0] += 1.0
```

```
            if t1[i] == '0' and t2[i] == '0':
                sim[2] += 1.0
            if t1[i] == '1' and t2[i] == '0':
                sim[1] += 1.0
            if t1[i] == '0' and t2[i] == '1':
                sim[1] += 1.0
        #Three different methods for computing
        if n == 'S': #SMC
            result =  (sim[0] + sim[2])/sum(sim)
        if n == 'J':  #JACCARD
            result =  sim[0]/(sim[0]+sim[1])
        if n == 'D': #DICE
            result = (2*sim[0])/(2 * sim[0] + sim[1])
        return result

    t_data = dataforp
    result_list = ""
    test_attr = csv.reader(tf)
    learn_attr = []
    learn_class = []
    #To learn the data
    for key_m in t_data:
        for key_n in t_data[key_m]:
            learn_attr.append(key_n)
            learn_class.append(key_m)

    #Begin predicting
    for attr in test_attr:
        #compute the distances or similarities according to input
        temp_distance = []
        for lis in range(len(learn_class)):
            if distance_measure == 'similar':
                temp_distance.append(similar(learn_attr[lis],attr,'D'))

        if distance_measure == 'distance':
            temp_distance.append(distance(learn_attr[lis],attr))
        #get the sorted indices of distances list
        sortedindex = array(temp_distance).argsort()
        h = 0
        #Begin to find the nearest neighbour(s) according to input K
        while h < k:
                classcount = {}
        #select the smallest distance or most similar neighbour(s) for voting the prediction
                if distance_measure == 'similar':
                        voteclass = learn_class[sortedindex[::-1][h]]
                if distance_measure == 'distance':
                        voteclass = learn_class[sortedindex[h]]
                classcount[voteclass] = classcount.get(voteclass,0) + 1
                h = h+ 1
        maxcount = 0
        #select the answer which are voted the most
        for key1, value in classcount.items(): #majority voting
                if value > maxcount:
                        maxcount = value
                        maxindex = key1
        answer = maxindex
        result_list += answer
end = time.clock()
print "K-NN classifier, K = %s, using %s measure" % (sys.argv[2],sys.argv[1])
print "Processing time: %fs" % (end-start)
print "Prediction: %s" % result_list
f.close()
tf.close()
```