# Hunting Brand Domain Forgery: A Scalable Classification for Homograph Attack

Tran Phuong Thao[1(✉)], Yukiko Sawaya[1], Hoang-Quoc Nguyen-Son[1], Akira Yamada[1], Kazumasa Omote[2], and Ayumu Kubota[1]

[1] KDDI Research, Inc.,Fujimino, Japan
{th-tran,yu-sawaya,ho-nguyen,ai-yamada,kubota}@kddi-research.jp
[2] Tsukuba University,Tsukuba, Japan
omote@risk.tsukuba.ac.jp

**Abstract.** Visual homograph attack is a way that the attackers deceive victims about what domain they are communicating with by exploiting the fact that many characters look alike. The attack is growing into a serious problem and raising broad attention in reality when recently many brand domains have been attacked such as `apple.com` (Apple Inc.), `adobe.com` (Adobe Systems Incorporated), `lloydsbank.co.uk` (Lloyds Bank), etc. Therefore, how to detect visual homograph becomes a hot topic both in industry and research community. Several existing papers and tools have been proposed to find some homographs of a given domain based on different subsets of certain look-alike characters, or based on an analysis on the registered International Domain Name (IDN) database. However, we still lack a scalable and systematic approach that can detect sufficient homographs registered by attackers with a high accuracy and low false positive rate. In this paper, we construct a classification model to detect homographs and potential homographs registered by attackers using machine learning on feasible and novel features which are the visual similarity on each character and some selected information from Whois. The implementation results show that our approach can bring up to 95.90% of accuracy with merely 3.27% of false positive rate. Furthermore, we also make an empirical analysis on the collected homographs and found some interesting statistics along with concrete misbehaviors and purposes of the attackers.

**Keywords:** Web security · International domain name · Punycode · Visual homograph attack

## 1 Introduction

Visual homograph attack was first described by Gabrilovic [1]. To prove the feasibility of this kind of attack, the authors registered a homograph of the brand


Published by Springer Nature Switzerland AG 2019
G. Dhillon et al. (Eds.): SEC 2019, IFIP AICT 562, pp. 3–18, 2019.
https://doi.org/10.1007/978-3-030-22312-0_1

domain `microsoft.com` which incorporated Cyrillic characters. After that, several brand domains were targeted by homograph attacks but this attack was not much attracted. Until April 2017, when the `apple.com` was forged by the homographs [2] such as the one appears under Punycode form `xn-pple-43d.com` which uses the Cyrillic '`а`' (U+0430) instead of the ASCII '`a`' (U+0061), the attack got mass attention from the media, and thus how to detect visual homographs becomes a significant issue.

Right after the attack on `apple.com` was published, some web browsers disabled the function of automatic IDN conversion. Since the IDNs contain non-ASCII characters (e.g., Arabic, Chinese, Cyrillic alphabet), they are encoded to ASCII strings using Punycode transcription known as *IDNA* encoding and appear under ASCII strings starting with `xn--`; for example, `xn--ggle-0qaa.com` is displayed as `gõõgle.com`. However, there is a big trade-off when a web browser stops supporting the automatic IDN conversion because a huge number of Internet users are using non-English languages with non-Latin alphabets through over 7.5 million registered IDNs in all over the world (by December 2017) [3]. Furthermore, visual homograph not only takes advantage of look-alike Punycode characters in IDNs, but also look-alike Latin characters in even non-IDNs themselves; for example, the homograph `bl0gsp0t.com` was registered targeting to the brand domain `blogspot.com` by replacing '`o`' by '`0`', or the homograph `wlklpedia.org` was registered targeting to the brand domain `wikipedia.com` by replacing '`i`' by '`l`'. Also, if homograph domains can deceive users before they appear in the address bar of web browsers (e.g., homographs are given from an email or a document under hyper-links) without the users' awareness of the browsers, disabling IDN conversion is not meaningful to prevent users from accessing the homographs. Therefore, the web browsers after that re-enabled the function but are trying to block homographs which can be detected or blacklisted. Then, the problem is still how to detect homographs. Several existing tools and previous papers such as [4–10,17] have been proposed to find homographs of given domains using different inadequate subsets of certain look-alike characters that are defined by themselves, or using the IDN database registered at the time of analysis.

Therefore, our goal is how to propose a scalable, systematic, high-accuracy and low-false-positive-rate approach that can detect sufficient visual homographs not registered by the brand domains' owners to pro-actively protect their brands but by attackers. The research scope in this paper is described as follows. First, there are several types of homographs such as visual-based, semantic-based, top-level-domain (TLD)-based, typosquatting-based but this paper focuses only on the visual-based homograph which is the most popular and serious type (The explanation and the reason why visual-based homograph is the most serious will be described in more details in the background). Second, this paper focuses on finding homographs registered by attackers that can be either phishing (being active and having phishing content) or not phishing yet (being active and not yet have phishing content, but we still consider it as harmful case because of the behavior of registering homograph targeting to brand domains); in other words, our aim is not to detect phishings but homographs.

### 1.1 Related Work

Many existing tools such as [4–7], or previous work such as [8] by Abawajy et al. have been proposed to find visual homographs on inputting a (brand) domain. Most of these tools simply define a subset of look-alike characters, and then replace each character in the given domain by the look-alike characters in the subset. Some of them such as [7] look up Domain Name System (DNS) to determine whether the homographs are active or not. However, using the tools and previous works cannot find sufficient homographs because the subsets are too small compared with the enormous set of look-alike characters. Furthermore, their approaches cannot distinguish which homographs are registered by owners of the brand domains (to protect their brands), and which homographs are registered by attackers. Also, a formal measure (e.g., Structural Similarity Index (SSIM), Peak Signal-to-Noise Ratio (PSNR) or Mean Squared Error (MSE), etc.) is not used to define the visual similarity instead of subjective feelings on the visual look. There are also some popular tools such as [11,12] but note that the tools are used to generate the other types of homographs like TLD-based or typosquatting, not visual-based as our goal. Tian et al. [13] proposed a method to predict phishing homographs by analyzing HTML content and visual screen-shots. However, using malicious HTML content and similar visual screen-shots does not mean that the phishing has to be homograph in the domain name string. For example, `random.com` (non-homograph) and `faceböök.com` (homograph) are both phishing to the brand *facebook.com* but the former one is a non-homograph and the latter one is a homograph. In other words, the features (i.e., HTML content and screen-shot) are reasonable for detecting phishings but not homographs in term of domain string. Moreover, downloading HTML content and capturing screen-shots require the analyzers to access the domains, and that can lead to malware injections or the cost for setting up a virtual machine. Oliver et al. [10] detect illegitimate links including homographs on a web page but it is not clear which criteria they use to formally define about visual similarity. In their work, they describe "One solution is to introduce a table of characters (i.e., glyphs) that are considered visually similar"; therefore probably their approach is similar to [4–8] as we mentioned above. The same issue is in the work by Tyson et al. [17]. Most recently, Liu et al. [9] have been proposed to detect homographs by using a visual similarity metric for the images of entire domain strings between each of 1.4 million registered IDNs and each of top 1000 popular domains ranked by Alexa. The drawback is that, applying the visual similarity for entire domain strings can lower the accuracy and lead to high false positive rate. When we analyzed their dataset, we found that many domain pairs of the brand domains and sample domains that are too different but have very high SSIM on the entire domain strings; for example, à`a.com` and `ea.com` have 0.952 SSIM and are listed as homographs but actually not. Furthermore, the paper only considers IDNs but as mentioned above, homographs can occur even in Latin alphabet such as ('`I`', '`1`', '`l`') or ('`o`', '`0`'). Also, lots of new domains are registered everyday and thus the method is not scalable.

### 1.2 Our Work

In this paper, we propose a machine learning based approach which meets the following contributions:

- To the best of our knowledge, ours is the first study proposing a classification of homographs and non-homographs using a visual similarity measure (i.e., SSIM) on each character instead of entire domain string as previous work. This approach can increase the accuracy to 95.36% with merely 2.83% of FPR.
- Although using Whois for detecting phishing is not a new approach, but when it is combined with SSIM on each character, the approach becomes promising to detect homographs and eliminate the domains that look alike to the brand domains but are registered by the brand domains' owners. We do not trivially use entire Whois but select the practical features such as: creation date of homograph is often after that of the brand domain, expiration date of homographs is often before that of the brand domain, register name, and organization of homograph is different from that of the brand domain, along with original creation date and expiration date of homograph and brand domain. The evaluation result shows that our approach can reach to 95.90% of accuracy with merely 3.27% of FPR.
- Last but not least, we make an empirical analysis on the collected homographs and found that a large portion of the domains (44.57%) are for sale or parked domains, 6.38% have the same/related content to the brand domains, 32.45% cannot be accessed or have blank content, 2.13% were created as an education about what is homograph, and 14.47% have completely different content with the brand domains. Interestingly, we figured out several concrete misbehaviors and purposes of the hackers when analyzing these homographs.

Note that, the accuracy when using SSIM only is 95.36% and when using SSIM with Whois is 95.90%. It does not mean that using Whois does not bring much effect (increasing only 0.54% of accuracy because the processes of data labeling in the two cases are different. The important thing here is the high accuracy (over 95%) in both cases.

### 1.3 Roadmap

The rest of this paper is organized as follows. The backgrounds of homograph attacks, visual similarity measure, and Whois are described in Sect. 2. Our proposed method is presented in Sect. 3. The experiment results are analyzed in Sect. 4. The empirical analysis on the labelled homographs is described in Sect. 5. The discussion of several ideas for future work is given in Sect. 6. Finally, the conclusion is drawn in Sect. 7.

## 2 Backgrounds

In this section, we present the backgrounds of homograph attacks, visual similarity measures in which SSIM is used in this paper, and Whois information.

### 2.1 Homograph Attacks

Homograph attack is a way that the attackers deceive victims about what domain they are communicating with by exploiting the fact that many domains look alike. There are several kinds of homographs in the wild, we thus synthesize them into five categories. The first is **visual homograph** which uses different but visually look-alike characters, for example: `facebook.com` and `faceböök.com`. The second is **semantic homograph** which uses synonyms or contextual similar words, for example: `facebook.com` and `mark_zuckerberg_social_network.com`. The third is **TLD homograph** which uses the same main domain names, but different top-level-domain (TLD), for example: `facebook.com` and `facebook.biz`. The fourth is **typosquatting** which relies on mistakes such as typos made by Internet users when typing the domain names, for example: `facebook.com` and `faceboook.com`. The last is the combination of the previous 4 categories. Note that the homographs in which certain characters are inserted or replaced (known as bitsquatting) in the brand domains are also listed in the fourth type (typosquatting homograph); for instance, `travelgoogle.com` targeting to `google.com`. In this paper, we focus on the first that is visual homograph since it is the most popular and serious type.

*Visual Homograph Attack.* Why visual homograph is the type that is serious the most? First, only the visual homograph can produce a fake domain that is 100% look-alike with the brand domain and even human cannot distinguish. For example, the brand domain `google.com` and the visual homograph `google.com` (encoded by `xn--gogle-m29a.com`) which are completely look-alike with the brand domain, so have very high probability to deceive users. Second, visual homograph not only utilizes the look-alike Punycode characters but also even the look-alike Latin characters such as '`I`' (big `i`), '`l`' (el) or '`1`' (one). For example, the visual homograph `ad0be.com` targeting to the brand domain `adobe.com` by replacing '`0`' by '`o`'.

### 2.2 Visual Similarity Measure

Visual similarity is a method for measuring the similarity between two images. In this paper, we use the state-of-the-art metric called Structural Similarity Index (SSIM) [14] which is perceptual measure based on visible structures in the images. Meanwhile, the traditional methods such as Peak Signal-To-Noise Ratio (PSNR) and Mean Squared Error (MSE) estimate absolute errors only. The SSIM between two images $x$ and $y$ of the same size $N \times N$ is:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{1}$$

$\mu_x$ and $\mu_y$ represent the averages of $x$ and $y$ respectively. $\sigma_x^2$ represents the covariance of $x$ and $y$. $\sigma_x^2$ and $\sigma_y^2$ represent the variances of $x$ and $y$ respectively.

| | |
|---|---|
| [Domain Name] | KDDI.JP |
| [Registrant] | KDDI CORPORATION |
| [Name Server] | dns101.dion.ne.jp |
| [Name Server] | dns102.dion.ne.jp |
| [Name Server] | dnsa01.kddi.ne.jp |
| [Name Server] | dnsa02.kddi.ne.jp |
| [Creation on] | 2001/04/16 |
| [Expires on] | 2017/04/30 |
| [Status] | Active |
| [Last Updated] | 2016/05/01 01:05:12 (JST) |
| Contact Information: | |
| [Name] | KDDI CORPORATION |
| [Email] | kt-tanaka@kddi.com |
| [Web page] | |
| [Postal code] | 163-8003 |
| [Postal Address] | 3-2, Nishishinjuku 2-chome, Shinjuku-ku |
| [Phone] | 03-3347-5818 |
| [Fax] | |

**Fig. 1.** An example of Whois: the Whois of the domain "kddi.jp"

$c_1 - (k_1 L)^2$ and $c_2 - (k_2 L)^2$ represent the variables to stabilize the division with weak denominator where $L$ is the dynamic range of the pixel-values and is typically set to $L = 2^{\#bits\_per\_pixel} - 1$ and $k_1 = 0.01, k_2 = 0.03$ by default. SSIM values $[-1, 1]$ where 1 indicates perfect similarity.

### 2.3 Whois

Whois [15] is a protocol that is used for querying databases that store the information of the registered domains such as domain name, registrar, creation date, expiration date, organization, email, etc. Nowadays, there are many competitive services supporting Whois queries by web portal or API. An example of Whois is given in Fig. 1.

## 3 Our Proposed Method

In this section, we present our method including data collection, data labelling, feature extraction and selection, and learning process.

### 3.1 Data Collection

The most adequate method to collect homographs is to use the Confusable Unicode table [16] defined by Unicode, Inc. This table (for example version 11.0.0 updated on 2018-05-25) contains 6,296 pairs of confusable characters. However, it is impossible since using the entire set of the confusable pairs, then get all

permutations for each position in the domains, and finally query Whois for each permutation are too inefficient (also, the number of non-homographs are extremely dominant compared with the number of homographs). Instead, the way we collected homographs is as follows.

First, in 6,296 Unicode confusable pairs mentioned above, we use only the pairs that have the targeting characters such as A–Z, 1–9 and the hyphen (-) because most of the brand domains (top Alexa ranking) are non-international-domain-name (non-IDN) and thus contains these characters only. We generated 26,021 homographs, then queried their Whois but only got 37 registered domains. Second, we use some tools in the wild such as [4–7] to generate 12,338 homographs which are different from the previous 26,021 generated ones (these tools use different subsets of similar characters), then we queried Whois and got 129 registered domains. Third, thanks to the authors of [9] for sharing us their 1,516 homographs that they matched 1,000 top popular domains ranked by Alexa with their 1.4 million registered IDN out of 300+ million registered domains; we then filter out the overlapping domains with the previous generated domains, re-queried Whois and got 1,006 registered domains (perhaps, at the time that we re-queried Whois, the other 510 domains already expired).

Totally, we got $39 + 129 + 1,006 = 1,174$ unique domains that are (at this time) called "temporary" homographs because we will annotate/label them again later. For non-homographs, from each of unique brand domains in the previous 1,174 "temporary" homographs, we generated at least one domain that is non-IDN and completely look different, and then got 1,969 domains. In summary, there are 3,143 domains in which 1,174 "temporary" homographs and 1,969 "temporary" non-homographs.

### 3.2 Data Labelling

Although the data we collected was "temporarily" labelled as homographs or non-homographs, we still need to re-label them again by the human because different persons will have different opinions about the visual homographs. For example, person A thinks `ess.com` is a homograph of `ass.com` but person B does not think that. For lowering the bias, we employed three analyzers to label the 3,143 domains with the same process and our final decision is based on majority rule. For example, 2/3 analyzers think `ess.com` is a homograph of `ass.com` so we finally label it as homograph. For different implementations below, we have different processes for data labelling:

*Case 1.* This case checks if using SSIM for each character can perform better than using SSIM for entire domain string. Thus, the three analyzers labelled them based on visual look only as Fig. 2. As a result, 3137 domains (99.81%) got the same label from the three analyzers; and the labels of the remaining 6 domains (0.19%) were decided based on majority (the labels decided by 2/3 analyzers). Finally, we got 1060 domains labelled as homograph and 2083 domains labelled as non-homograph.

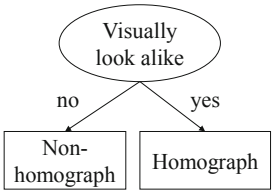


**Fig. 2.** Data labelling for case 1

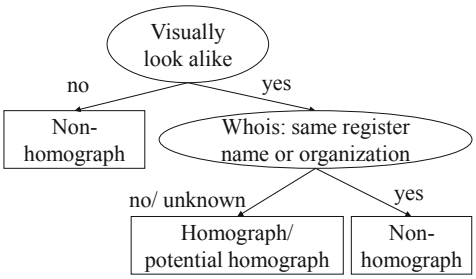


**Fig. 3.** Data labelling for case 2

*Case 2.* This case checks if using SSIM for each character and the Whois can classify the homographs (including potential homographs). The labelling process is described in Fig. 3. We consider potential homographs here because the register names and organizations of some domains (even brand domains or sample domains) are hidden to protect the privacy of domain owners' information. For such pairs of the brand and sample domains that have unknown/hidden register name and organization, we treat them as potential homographs. Also, we cannot separate these potential homographs from actual homograph because we cannot confirm that they are actual or potential homographs. In the case 1, the visual looks are different from each person, the data labelling was thus done by the human. However, in this case 2, the Whois information is obvious, we thus only re-use the labelling result from case 1 and create a program to combine it with the condition of whether the register name and organization are different. In total 3143 domains, we finally got 940 domains labelled as homograph/potential homograph, and 2203 domains labelled as non-homograph.

### 3.3 Feature Extraction and Selection

For each case mentioned in the data labelling, the process of feature extraction and selection are described as follow:

*Case 1.* For each pair of the sample domain and its brand domain, we compute SSIM for each character and get the average. More concretely, we first separate the domains into each character. For each pair of characters in order, we parse them into images and compute SSIM for the images. Finally, the average of all SSIMs for every pair of characters which is corresponding to every position in the domain string. For example, SSIM of the two domains '`ab.jp`' and '`xy.vn`' is the average of the following five pairs: SSIM(image('`a`'), image('`x`')), SSIM(image('`b`'), image('`y`')), SSIM(image('`.`'), image('`.`')), SSIM(image('`j`'), image('`v`')), and SSIM(image('`p`'), image('`n`'). For the pairs of homographs and brand domains that have a different number of characters (domain string lengths), they are labelled as non-homograph in Sect. 3.2 without the need to compute the SSIM. There are a few examples when the homographs and brand

domains have a different string such as '`rn`' and '`m`' but they are very rare so we do not consider.

*Case 2.* For this case, besides the SSIM computed on each character in the domain strings as the first case, the Whois of each brand and sample domains is also queried. More concretely, we extract the register name, organization, creation date, and expiration date. We finally use the following 15 features: (1) the average of SSIM on each character; (2) whether register of the brand domain is different from that of sample domain (1 if yes, 0 if no and 2 if both are none or hidden, we do not need to consider the case when one of them (not both) is none or hidden because as long as they are different, they cannot be non-homographs); (3) whether organization of the brand domain is different from that of sample domain (the values are the same as (2)); (4) whether creation date of the brand domains is before that of sample domains (1 if yes and 0 if no); (5) whether expiration date of the brand domain is after that of sample domains (1 if yes and 0 if no); (6) creation year of the brand domain; (7) creation month of the brand domain; (8) expiration year of the brand domain; (9) expiration month of the brand domain; (10) lifetime of the brand domain (the number of days between creation date and expiration date); (11) creation year of sample domain; (12) creation month of sample domain; (13) expiration year of sample domain; (14) expiration month of sample domain; and (15) lifetime of sample domain (the number of days between creation date and expiration date).

### 3.4 Learning

Since the homograph/non-homograph samples are collected consecutively for each given brand domain, the data must be randomly shuffled at first in order for reducing variance and making sure that models remain general and overfit less. Then, we apply 7 popular supervised machine learning algorithms for training process including Support Vector Machine, Naive Bayes, Decision Tree, Neural Network, Stochastic Gradient Descent, Nearest Neighbors and Logistic Regression. We use $k$-fold cross validation ($k$ is set to 10 in our implementation) and compute the accuracy, false positive rate, and true positive rate for our model. Also, the ROC curves are drawn to depict the comparison of previous and our approaches.

## 4 Experiment

The programs written in Python 2.7.11 on a computer Intel(R) core i7, RAM 16.0 GB, 64-bit Windows 10. The Whois is extracted using the *python-whois* package version 0.6.3. The machine learning algorithms are applied using *scikit-learn* package version 0.18. The SSIM is computed using the *skimage* package version 0.15.dev0.

### 4.1 Parameters

For each model, different parameters are used. For the Support Vector Machine (SVM), 3 parameters used are SVC, NuSVC which support different kernels and LinearSVC which supports only a linear kernel. For the Naive Bayes, 3 parameters used are GaussianNB which implements the Gaussian Naive Bayes, MultinomialNB which implements the Naive Bayes for multinomially distributed data, and BernoulliNB which implements the Naive Bayes for data distributed according to multivariate Bernoulli distributions. For the Nearest Neighbors, 3 parameters used are KNeighborsClassifier which implements learning based on the n_neighbors nearest neighbors of each query point, n_neighbors = 5 is set by default, RadiusNeighborsClassifier which implements learning based on the number of neighbors within a fixed radius of each training point, radius = 1.0 is set by default), and NearestCentroid which represents each class by the centroid of its members. For the Decision Tree, only 1 parameter used is DecisionTreeClassifier (note that Decision Tree has several algorithms such as ID3, C4.5, CART, CHAID, MARS, and Conditional Inference Tree but only the optimized version of the CART is used in this experiment. For the Neural Network, only 1 parameter used is MLPClassifier which implements a multi-layer perceptron that trains using Backpropagation. For the Stochastic Gradient Descent, only 1 parameter used is SGDClassifier which implements a plain stochastic gradient descent learning routine which supports different loss functions and penalties for classification. Finally, for the Logistic Regression, only 1 parameter used is LogisticRegression.

### 4.2 Results

The result for each case is described as follows:

*Case 1.* The results are described in Table 1. For the approach of using SSIM on the entire string, the best accuracy is 86.73% (with FPR = 5.41%) performed by KNeighborsClassifier. For our approach of using SSIM on each character, the best accuracy is 95.35% (with FPR = 2.83%) performed also by KNeighborsClassifier. We achieve 8.62% higher accuracy and 2.58% lower FPR than the previous approach. In this case, only the SSIM is used as the feature and what we expect is that the samples are classified as homograph if its SSIM is larger than a threshold and vice versa. There is only KNeighbors algorithm that classifies an object by a majority vote of its neighbors, with the object being assigned to the class most common among its $k$ nearest neighbors. That is why the KNeighborsClassifier performs the best. The ROC curves of our and previous approaches are depicted in Fig. 4. The light curves mean the curves in each of $k = 10$ folds and the unique bold curve means the average for all the folds.

*Case 2.* In this case, we tried almost the same parameters for each algorithm as the case 1 except only the RadiusNeighborsClassifier. Since the value of year is much larger than the other features' values, the radius is set larger (i.e., 1400) to

avoid the case when some outlier samples do not have any neighbor within the given radius. The results are described in Table 2. Note that, we implemented the case of SSIM on the entire string with Whois for fairly comparing with our achievement that is using SSIM on each character with Whois; there is no previous work using SSIM on the entire string with Whois but SSIM on the entire string only. The result shows that, for the approach of using SSIM on the entire string with Whois, the best accuracy is 92.43% (with FPR = 6.01%) performed by DecisionTreeClassifier. For our approach of using SSIM on each character with Whois, the best accuracy is 95.90% (with FPR = 3.27%) performed also by DecisionTreeClassifier. In this case 2, the way we label the samples is using SSIM at first to filter out the samples that are look alike to the brand domains, and if some domains satisfy this condition, the Whois is then used to filter out the ones which have the same owner with the brand domains. Therefore, it is reasonable why DecisionTree can perform the best because only this algorithm works based on decision rules in a flowchart structure. The ROC curves of the two approaches are depicted in Fig. 5. The explanation for the light curves and the unique bold curves is the same as that in case 1 above.

*Why the ROC Curves Strictly Change Direction at the Cut-Point.* In both case 1 and 2 and also in both case of using SSIM on each character and case of using SSIM on the entire domain string, the curves do not increase regularly with the same acceleration, but strictly change direction at the cut-point of each curve. This is because applying SSIM on text for a classification has only two groups which are when the SSIM is under a threshold and when the SSIM is over the threshold. It is different from applying SSIM on more complex images (such as landscape or person portrait) which has multiple groups because other elements are considered such as the color grayscale (the amount of light or color intensity).

## 5 Empirical Analysis on Labelled Homographs in Case 2

In this section, we make an empirical analysis on 940 domains labelled as homograph/potential homograph in the case 2 and found the following results. 60 domains (6.38%) that have the same or related content to the brand domains. 305 domains (32.45%) cannot be accessed or have blank content. 20 domains (2.13%) were created as an education about what is homograph. 419 (44.57%) domains are for sale or parked domains (Parked domains are a kind of domains that are registered without being associated with any services. Instead, these idle domains are used to display relevant advertisements; and every time a consumer clicks on one of the advertisements, the owner can earn money). Finally, 136 domains (14.47%) have completely different content compared with that of the brand domains. Interestingly, while analyzing these 136 domains, we figure out some unusual misbehaviors of the hackers:

– Several homographs were created not for any harmful purpose but just for saying some pointless words. For example, `xn--pple-koa.com` (displayed as ä`pple.com`) targeting to the brand domain `apple.com` has a web content like "the art of killing yourself" or "why is everybody so stupid".

- Several homographs that redirect user's accesses to a safe webpage for the purpose of selling product only. For example, `xn--yotbe-lvab.com` (displayed as `yoütübe.com`) redirects user to a page on `amazon.com` that is selling a Kindle E-reader.
- Several homographs were created with the purpose of increasing pageview. For example, `xn--youtbe-6ya.com` (displayed as `youtübe.com`) is just a music video on Youtube with almost 5 million views and 2.3 thousand subscribers. This is also a way to make money (Youtube pays users based on the number of video views).
- Several homographs were created with the purpose of advertising the hackers themselves. For example, `xn--facebok-q0a.com` (displayed as `faceboók.com`) describes a hacker's profile with a message "If you'd like to hire me ...".
- Several homographs were created to claim (or lower the reputation of) the brand domains for the hacker's demands. For example, `xn--microsftnline-1kdc.com` targeting to `microsoftonline.com` claims Microsoft to support Cyrillic alphabet in its keyboard.

**Table 1.** Evaluation result for case 1

| Algorithms | SSIM on entire string | | | SSIM on each character | | |
|---|---|---|---|---|---|---|
| | Acc(%) | FPR(%) | TPR(%) | Acc(%) | FPR(%) | TPR(%) |
| svm.SVC | 84.98 | 10.62 | 76.35 | 94.18 | 6.6 | 95.64 |
| svm.NuSVC | 85.14 | 10.38 | 76.35 | 94.02 | 6.83 | 95.64 |
| svm.LinearSVC | 86.35 | 8.37 | 75.96 | 94.21 | 5.9 | 94.45 |
| GaussianNB | 84.98 | 10.62 | 76.35 | 94.21 | 6.55 | 95.64 |
| MultinomialNB | 66.27 | 0.00 | 0.00 | 66.27 | 0.00 | 0.00 |
| BernoulliNB | 66.27 | 0.00 | 0.00 | 66.27 | 0.00 | 0.00 |
| NearestCentroid | 82.06 | 15.62 | 77.47 | 92.91 | 8.75 | 96.07 |
| KNeighbors | **86.73** | **5.41** | **71.22** | **95.35** | **2.83** | **91.79** |
| RadiusNeighbors | 66.27 | 0.00 | 0.00 | 66.27 | 0.00 | 0.00 |
| DecisionTree | 82.18 | 13.72 | 74.04 | 94.85 | 4.05 | 92.61 |
| MLPClassifier | 86.29 | 8.03 | 75.14 | 94.21 | 5.28 | 93.21 |
| SGDClassifier | 81.39 | 13.63 | 70.84 | 92.02 | 8.95 | 94.1 |
| LogisticRegression | 66.27 | 0.00 | 0.00 | 66.27 | 0.00 | 0.00 |

*Abbreviation: Acc (accuracy), FPR (false positive rate), TPR (true positive rate)*

## 6 Discussion

This section describe several challenges for future work to improve the accuracy.

*Average SSIM for Different Characters and Other Measures for SSIM.* In this paper, we compute the average SSIM for all the characters. For example, for the pair of domains `foo.jp` and `föö.jp`, the SSIM currently used is the average of

**Table 2.** Evaluation result for case 2

| Algorithms | SSIM on string + Whois | | | SSIM on character + Whois | | |
|---|---|---|---|---|---|---|
| | Acc(%) | FPR(%) | TPR(%) | Acc(%) | FPR(%) | TPR(%) |
| svm.SVC | 82.12 | 2.64 | 46.28 | 82.12 | 2.64 | 46.28 |
| svm.NuSVC | 81.36 | 1.69 | 41.48 | 81.36 | 1.69 | 41.48 |
| svm.LinearSVC | 70.44 | 19.74 | 45.41 | 68.04 | 24.28 | 50.55 |
| GaussianNB | 78.87 | 26.10 | 90.47 | 78.84 | 26.15 | 90.47 |
| MultinomialNB | 80.66 | 21.61 | 85.91 | 80.66 | 21.61 | 85.91 |
| BernoulliNB | 70.09 | 0.00 | 0.00 | 70.09 | 0.00 | 0.00 |
| NearestCentroid | 76.27 | 29.50 | 89.77 | 76.27 | 29.50 | 89.77 |
| KNeighbors | 85.27 | 10.01 | 74.08 | 85.27 | 10.01 | 74.08 |
| RadiusNeighbors | 85.08 | 11.25 | 76.41 | 85.08 | 11.25 | 76.41 |
| tree.DecisionTree | **92.43** | **6.01** | **88.68** | **95.90** | **3.27** | **93.93** |
| MLPClassifier | 79.22 | 18.35 | 74.03 | 78.66 | 14.85 | 62.89 |
| SGDClassifier | 79.32 | 9.02 | 51.90 | 76.74 | 22.74 | 75.09 |
| LogisticRegression | 84.63 | 12.38 | 77.61 | 84.63 | 12.38 | 77.61 |

*Abbreviation: Acc (accuracy), FPR (false positive rate), TPR (true positive rate)*

SSIM of each pair (‘f’, ‘f’), (‘o’, ‘ö’), (‘o’, ‘ö’), (‘.’, ‘.’), (‘j’, ‘j’), (‘p’, ‘p’). However, for the domains that have a small number of different characters compared with the number of all characters, using average SSIM cannot reflect clearly the visual difference; instead, the average SSIM on the different characters only perhaps can improved the accuracy. Concretely, a promising SSIM is the average of SSIM of two pairs (‘o’, ‘ö’), (‘o’, ‘ö’). Furthermore, besides using the *average* for SSIM, other measures such as the *median* (the middle value of a range), the *covariance* (the expected value of variations of two random variates from their expected values), or the *correlation* (the expected value of two random variates) can be considered.

*Additional Features.* Several other features may also help. First, homograph domains have the search engine's result count is less than that of the brand domains. Second, homograph domains often target to the brand domains that are hot topics such as crypto-currency or payment. We analyzed a dataset of phishing sites downloaded from PhishTank, and found that the top three categories with a dominant number of homographs are crypto-currency (38.3%), payment system (20.7%) and online game (10.2%), not bank or online shopping as we thought. Third, the time distance between the creation date/expiration date in the Whois from the present should be also considered as the features because the real brand domains have longer age (as the present) than the homograph domains.

*Removing Diacritical Marks.* Diacritical marks (also diacritical sign or accent) is a glyph added to a letter. For example, `yáhoo.com` (encoded by `xn--yhoo-5na.com`, targeting to the brand domain `yahoo.com`) have the dia-

critical acute mark ‘´’ over the letter ‘a’. By removing the diacritical marks as a pre-process before data labelling, the accuracy may be improved but probably it can also lead to higher false positive rate; thus a re-implementation is necessary.

*Other Implementations.* The implementation of the case 2 should be divided into two sub-implementations. The first one uses only SSIM and the second one uses both SSIM and Whois as the current implementation even though they have the same data-labelling process. These two sub-implementations can help figure out how **fairly effective** when Whois is used without comparing it to the implementation of case 1. Furthermore, even though SSIM is proven to be better than the traditional measures such as PSNR and MSE, an additional implementation should be done to confirm whether our approach when using SSIM outperforms it when using PSNR or MSE.

*Extending Research Scopes.* This paper currently deals with visual homograph but how to thoroughly deal with all types of homographs described in Sect. 2.1 becomes a great demand. The TLD-based is the most trivial since we can straightforwardly replace the TLD with the un-large entire set of available TLDs (1,535 TLDs). The typosquatting may be not difficult since there are several tools that can be used to create the samples such as [7,11,12]. However, the semantic-based is the most challenge since as far as we know, there is no existing methods which can automatically collect enough a number of samples.

*Whether Homographs Are Caused By Fonts?* Someone may question that if we compare characters in Arial and Times, the characters are probably slightly different. However, homographs are not caused by the font differences but Unicode code differences. Even if ‘A’ in Arial and Times fonts are visually look different, they have the same Unicode code that is “U+0041”. Furthermore, in any web browser, the font (and font size) can be changed in the web body interface but not in the address bar.

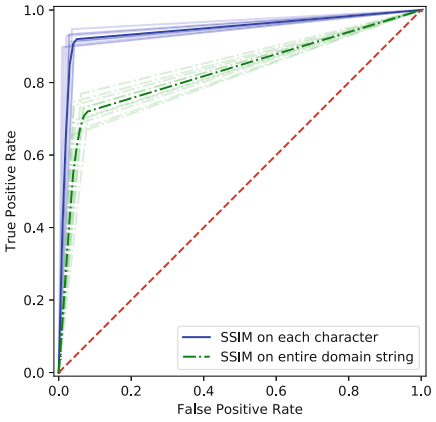


**Fig. 4.** ROC curves in case 1

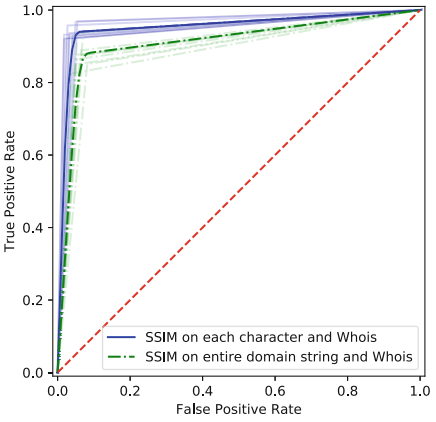


**Fig. 5.** ROC curves in case 2

## 7 Conclusion

This paper proposes the first classification of homographs registered by attackers and non-homographs by using the state-of-the-art visual similarity metric that is SSIM on each character along with some reasonable selected information from Whois to increase the accuracy up to 95.90% with merely 3.27% of FPR. Two implementation cases are analyzed to explain how the approach works when applying only SSIM and applying the combination of SSIM and Whois (with the different labelling processes). An empirical analysis on labelled homographs is also taken place to find the ratio between different contents of the homographs, and to understand certain concrete purposes of the attackers.

**Acknowledgement.** This research was carried out as part of WarpDrive: Web-based Attack Response with Practical and Deployable Research InitiatiVE, the Commissioned Research of the National Institute of Information and Communications Technology (NICT), JAPAN.

## References

1. Gabrilovic, E., Gontmakher, A.: The homograph attack. Commun. ACM **45**(2), 128 (2002)
2. Xudong, Z.: Phishing with Unicode Domains (2017). https://www.xudongz.com/blog/2017/idn-phishing/?_ga=2.53371112.1302505681.1542677803-1987638994.1542677803
3. IDN World Report: Internationalised Domains show negative growth in 2017. https://idnworldreport.eu/
4. Idn-homograph-attack. https://github.com/timofurrer/idn-homograph-attack
5. EvilURL. https://github.com/UndeadSec/EvilURL
6. Homographs. https://github.com/dutchcoders/homographs
7. Dnstwist. https://github.com/elceef/dnstwist
8. Abawajy, J., Richard, A., Aghbari, Z.A.: Securing websites against homograph attacks. In: Lin, X., Ghorbani, A., Ren, K., Zhu, S., Zhang, A. (eds.) SecureComm 2017. LNICSSITE, vol. 239, pp. 47–59. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-78816-6_4
9. Liu, B., et al.: A reexamination of internationalized domain names: the good, the bad and the ugly. In: 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2018) (2018)
10. Hunt, O.J., Krstic, I.: Preventing URL Confusion Attacks. Patent US9203849B2 United States. Apple Inc. (2013)
11. Instant Domain Search: Domain Name Generator. https://instantdomainsearch.com/domain/generator/
12. DN Pedia: Search Domain Zones. https://dnpedia.com/tlds/search.php
13. Tian, K., Jan, S., Hu, H., Yao, D., Wang, G.: Needle in a haystack: tracking down elite phishing domains in the wild. In: Internet Measurement Conference (IMC 2018), pp. 429–442 (2018)
14. Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)

15. Daigle, L.: WHOIS protocol specification. In: RFC 3912 Internet Society (2004). https://tools.ietf.org/html/rfc3912
16. Unicode Inc: Unicode Security Mechanisms for UTS #39. https://www.unicode.org/Public/security/latest/confusables.txt
17. Tyson, M., Peter, H., Greg, B.: The 2017 homograph browser attack mitigation survey. In: 15th Australian Information Security Management Conference. https://doi.org/10.4225/75/5a84f5a495b4d