Plagiarism Detection in Texts Obfuscated with Homoglyphs

Faisal Alvi,[a,b] Mark Stevenson,[a] and Paul Clough[a]

[a] University of Sheffield, Sheffield S10 2TN, United Kingdom,
[b] King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia.
{falvi1, mark.stevenson, p.d.clough}@sheffield.ac.uk

Abstract. Homoglyphs can be used for disguising plagiarized text by replacing letters in source texts with visually identical letters from other scripts. Most current plagiarism detection systems are not able to detect plagiarism when text has been obfuscated using homoglyphs. In this work, we present two alternative approaches for detecting plagiarism in homoglyph obfuscated texts. The first approach utilizes the Unicode list of confusables to replace homoglyphs with visually identical letters, while the second approach uses a similarity score computed using normalized hamming distance to match homoglyph obfuscated words with source words. Empirical testing on datasets from PAN-2015 shows that both approaches perform equally well for plagiarism detection in homoglyph obfuscated texts.

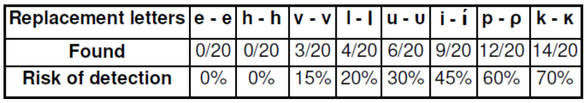
1 Introduction

The notion of 'Disguised Plagiarism' refers to a class of methods used for intentionally hiding text that has been copied [8]. Furthermore, 'Technical Disguise' is a particular form of disguised plagiarism, wherein obfuscation techniques are used in order to evade the detection of plagiarized text by changing the computational representation of text. An important method for technically disguising text is to substitute characters visually identical to other characters in some other script (i.e., homoglyphs) [5]. For example, the Latin character 'p' (Unicode U+160) and the Cyrillic 'р' (Unicode U+0440) have identical glyphs but distinct Unicode values, making the words 'paypal' and 'рayрal' appear identical to a human evaluator, but undetectable to an automated plagiarism detection system that has not been designed to deal with such changes. In tests of several leading plagiarism detection systems most were unable to detect similarities between source and plagiarized texts obfuscated using homoglyphs [7, 12].

In this work we present two alternate approaches for plagiarism detection in homoglyph obfuscated texts: (1) by using the Unicode list of 'confusables' to find and replace homoglyphs with visually identical ASCII letters; and (2) by using a measure of similarity based on normalized hamming distance to match homoglyph obfuscated words with source words. Our work shows both approaches perform equally well for detecting plagiarism in homoglyph obfuscated texts. Both approaches have their particular advantages and limitations and may therefore be applicable in specific application scenarios in homoglyph obfuscated texts.

2 Related Work

Homoglyph substitution has been used as part of standard tests for plagiarism detection systems. For example, Gillam et al. [5] used homoglyph substitution as an obfuscation strategy for testing plagiarism detection systems. Their results demonstrated that six out of seven plagiarism detection systems were unable to detect any similarity between source and substituted text. Figure 1 gives a list of characters used in their work, with the number of instances of each character visually detected by human evaluators stated as well. In the annual 'Plagiarism Detection Software Test' by Weber Wulff et al. [12], 13 out of 15 plagiarism detection systems failed to report any similarity between a given text source and its homoglyph substituted version.



Replacement letters	e - е	h - һ	v - ν	l - Ӏ	u - υ	i - í	p - ρ	k - κ
Found	0/20	0/20	3/20	4/20	6/20	9/20	12/20	14/20
Risk of detection	0%	0%	15%	20%	30%	45%	60%	70%

Fig. 1. Replacement Letters for Visually Similar Characters from [5]

Heather et. al [6] describe a variety of techniques for technically disguising plagiarized text, which include: modifying the character map, rearranging the glyphs in fonts, replacing text with graphical symbols, and inserting characters in background (white) font between words. Kakoneen and Mozgovoy [7] also discuss a number of 'technical tricks' that can be used to obfuscate texts, including: (1) the insertion of similar looking characters from foreign alphabets (homoglyph substitution); (2) the insertion of background colored characters in between spaces; and (3) the use of scanned images in place of text. According to their results, "*None of the evaluated systems were able to detect any instances of plagiarism from the documents.*"

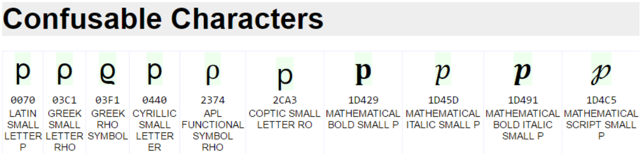
In addition to text obfuscation during plagiarism, homoglyphs have also been used in IDN (Internationalized Domain Name) homograph attacks[1] used to direct users towards alternative websites. With such an attack, users could be directed towards the website 'раураl.com' which is a Cyrillic substituted version of the Latin 'paypal.com'. Existing approaches to deal with IDN homoglyph attacks include: (1) Punycode [3] that converts non-ASCII characters into ASCII characters irreversibly (e.g., Ǵooǵle is converted to the ASCII 'xn–oole-ksbc'); (2) coloring-based strategies that distinguish homoglyphs by assigning various colors to foreign script characters [13]; and (3) a Unicode character similarity list (UC-SimList) [4] to detect homoglyphs in URLs. Some of these approaches might not be useful for plagiarism detection e.g. Punycode results in loss of information, and coloring requires visual inspection. However, the idea of using a list of Unicode equivalents for detecting IDN homograph attacks can be utilized for plagiarism detection in homoglyph obfuscated texts.

[1] `https://en.wikipedia.org/wiki/IDN_homograph_attack`

3 Methodology

3.1 Resources

The Unicode List of Confusable Characters. Several lists of homoglyph-alphabet pairs are freely available (e.g., `homoglyphs.net`). The Unicode consortium has released a list of *confusables*, which is a list of visually similar character pairs that includes homoglyphs and their corresponding Latin letters [1]. We use Version 9.0.0 of the list of confusables containing 6167 pairs of confusable characters. Figure 2 shows a partial list of letters similar to the letter ‘p’ taken from this list.



Confusable Characters

p	ρ	ϱ	р	⍴	ⲣ	𝐩	𝑝	𝒑	𝓅
0070	03C1	03F1	0440	2374	2CA3	1D429	1D45D	1D491	1D4C5
LATIN SMALL LETTER P	GREEK SMALL LETTER RHO	GREEK RHO SYMBOL	CYRILLIC SMALL LETTER ER	APL FUNCTIONAL SYMBOL RHO	COPTIC SMALL LETTER RO	MATHEMATICAL BOLD SMALL P	MATHEMATICAL ITALIC SMALL P	MATHEMATICAL BOLD ITALIC SMALL P	MATHEMATICAL SCRIPT SMALL P

Fig. 2. Visually Confusable characters for ‘p’ from the Unicode List of Confusables

Evaluation Dataset. We use PAN-2015 evaluation lab [11] dataset submission by Palkovskii and Belov [10] which is based on the PAN-2013 training dataset with characters in the suspicious documents replaced with homoglyphs. This dataset consists of 5185 document pairs divided into five categories of ‘no plagiarism’, ‘no obfuscation’, ‘translation’, ‘random’ and ‘summary’ obfuscation.

3.2 Approaches

Approach 1: Unicode Confusables. In our first approach (shown in Figure 3), we find and replace every non-ASCII character in the suspicious documents with the corresponding visually matching character from the list of confusable characters. This process replaces homoglyphs in the text of the suspicious documents with visually similar ASCII characters. The resulting suspicious documents can then be compared with the source documents for similarity. In our approach we use word trigram similarity as the seeding strategy, with merging and filtering to discard small matches as false positives [2].

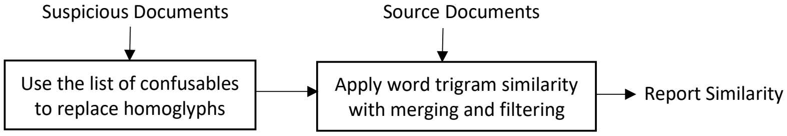

Suspicious Documents
Source Documents
Use the list of confusables to replace homoglyphs
Apply word trigram similarity with merging and filtering
Report Similarity

Fig. 3. Block Diagram for Plagiarism Detection using the List of Confusables

Approach 2: Normalized Hamming Distance. Our second approach uses normalized hamming distance as an approximate string matching technique. Such techniques are well-suited for this task since homoglyph substitution may partially change the structure of a word. *Hamming Distance* (when applied to strings of characters) detects the number of substitutions (replacements) from one string into another by finding the number of positions where the two strings differ [9]. We use a similarity score (sim_h) computed using normalized hamming distance, defined between two words w_1, w_2 of equal length as:

$sim_h(w_1, w_2) = 1 - \texttt{Number of substitutions}(w_1, w_2)/\texttt{length}(w_1)$.

Compared to other approximate string similarity measures, normalized hamming distance has the advantage of significantly reducing the number of false positives generated. Hamming distance is undefined for strings of unequal length, (we consider $sim_h = 0$ in this case), whereas these strings might be marked as similar using alternative string similarity techniques, such as character skip gram matching. For example, sim_h(play, plays) $= 0$, while sim_h(play, play) $= 0.75$.

Normalized hamming distance similarity (sim_h) is used to compare each word in the suspicious document with the words in the source document. If a pair of words have a value of sim_h greater than or equal to a particular threshold, we consider them as similar. The threshold value depends on the extent of homoglyph substitution in the dataset. For example, if most of the letters in each word have been replaced by homoglyphs, then a lower threshold value will be required to match these words.

Using this procedure for approximate matching of words instead of exact matching, we apply word trigram similarity with merging and filtering (as used in the list-based approach) to find the plagdet score between the source and suspicious documents. We conduct our experiments on the PAN-2015 dataset used in the list-based approach. Regarding the threshold value of sim_h for matching words in our experiments, we do not pre-select a value for this threshold. Instead we calculate plagdet scores for the entire dataset for a range of values of sim_h as shown in Figure 4.

4 Results and Discussion

4.1 Approach 1: Unicode Confusables

Table 1 shows the results of plagiarism detection in terms of Precision, Recall and Plagdet [11] scores. It can be seen that except for summary obfuscation, Plagdet scores for all other categories including that for the entire dataset are moderately high (≥ 0.60). This can be compared with the performance of most of the PAN approaches from 2012-2014 [11] on this dataset where the reported Plagdet scores were mostly 0, suggesting a significant improvement.

During the homoglyph replacement phase using the list-based approach, we observed that a number of replacements were also made for non-Latin characters in suspicious documents which were not intended as homoglyphs in source documents. For example, the currency symbol '¢' was replaced by a 'c'. This

Table 1. Plagdet-scores using the Homoglyph Replacement Approach

	Dataset	No Obf.	Random Obf.	Transl. Obf.	Summ. Obf.
Precision	0.772	0.663	0.953	0.781	0.826
Recall	0.727	0.988	0.667	0.643	0.107
Plagdet	**0.670**	**0.717**	**0.707**	**0.632**	**0.150**

observation suggests that the proposed approach of using a list of homoglyph-alphabet pairs to replace characters may not work well when the source text contains a large number of foreign characters, since these might be converted to ASCII characters in the substitution phase. However, the approach can be improved by searching through the source documents to distinguish homoglyphs from true source non-Latin characters, at the cost of increased computation time.

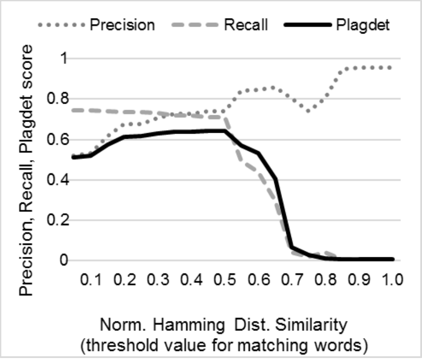


Fig. 4. Plagdet scores for Plagiarism Detection using Normalized Hamming Distance Similarity

Category	Plagdet
Entire Dataset	**0.644**
No Obfuscation	**0.662**
Random Obf.	**0.688**
Translate. Obf.	**0.626**
Summary Obf.	**0.142**

Table 2. Plagdet scores for the case when $sim_h = 0.450$

4.2 Approach 2: Normalized Hamming Distance

Figure 4 shows Precision, Recall and Plagdet scores for various values of sim_h threshold. It can be seen that a threshold value for $sim_h \approx 0.45$ is giving the highest Plagdet score of 0.644. Table 2 gives Plagdet scores for each category of plagiarism in the dataset for a threshold value of 0.45. Similar to Table 1, we observe that except for summary obfuscation, most of these values are moderately high (≥ 0.6). Although the scores in Table 1 are somewhat higher than those in Table 2, the differences are small enough to consider performance of the approaches to be similar for detecting plagiarism in homoglyph obfuscated texts. From Figure 4 we observe that a careful selection of threshold value for sim_h is important. The Plagdet score rapidly decreases after $sim_h = 0.5$ since higher threshold values increase the number of true matches being rejected. For large datasets, this problem can alleviated by first applying the approach on a smaller collection of training documents to obtain a suitable initial estimate for the threshold value.

5 Conclusions and Future Work

The development of techniques for automated plagiarism detection continues to be an active area of research. In this work we presented two approaches for plagiarism detection in homoglyph obfuscated texts which perform equally well for plagiarism detection. One approach utilizes the Unicode list of confusables to replace homoglyphs with visually identical letters; the other approach uses a similarity score computed using normalized hamming distance. For future work, improvised versions of these approaches can be incorporated into a set of approaches for detecting multiple forms of technical disguise.

References

1. Unicode List of Visually Confusable Characters. `http://www.unicode.org/Public/security/9.0.0/confusables.txt`, Online, Accessed: 2016-10-19
2. Alvi, F., Stevenson, M., Clough, P.D.: Hashing and Merging Heuristics for Text Reuse Detection. In: Working Notes for CLEF 2014 Conference. pp. 939–946 (2014)
3. Costello, A.: RFC3492-Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA). Network Working Group. `http://www.ietf.org/rfc/rfc3492.txt` (2003), Online: Accessed: 2016-10-19
4. Fu, A.Y., Deng, X., Wenyin, L.: REGAP: A Tool for Unicode-Based Web Identity Fraud Detection. Journal of Digital Forensic Practice 1(2), 83–97 (2006)
5. Gillam, L., Marinuzzi, J., Ioannou, P.: Turnitoff–Defeating Plagiarism Detection Systems. In: Proceedings of the 11th Higher Education Academy-ICS Annual Conference. Higher Education Academy (2010)
6. Heather, J.: Turnitoff: Identifying and Fixing a Hole in Current Plagiarism Detection Software. Assessment & Evaluation in Higher Education 35(6), 647–660 (2010)
7. Kakkonen, T., Mozgovoy, M.: Hermetic and Web Plagiarism Detection Systems for Student Essays An Evaluation of the State-of-the-Art. Journal of Educational Computing Research 42(2), 135–159 (2010)
8. Meuschke, N., Gipp, B.: State-of-the-Art in Detecting Academic Plagiarism. International Journal for Educational Integrity 9(1) (2013)
9. Navarro, G.: A Guided Tour to Approximate String Matching. ACM Computing Surveys (CSUR) 33(1), 31–88 (2001)
10. Palkovskii, Y., Belov, A.: Submission to the 7th International Competition on Plagiarism Detection. `http://www.uni-weimar.de/medien/webis/events/pan-15` (2015), Online, Accessed: 2016-10-15
11. Potthast, M., Göring, S., Rosso, P., Stein, B.: Towards Data Submissions for Shared Tasks: First Experiences for the Task of Text Alignment. In: Working Notes Papers of the CLEF 2015 Evaluation Labs. CEUR Workshop Proceedings (Sep 2015)
12. Weber-Wulff, D., Möer, C., Touras, J., Zincke, E.: Plagiarism Detection Software Test 2013 (2013), `http://plagiat.htw-berlin.de/software-en/test2013/report-2013/`, Online, Accessed: 2016-10-15
13. Wenyin, L., Fu, A.Y., Deng, X.: Exposing Homograph Obfuscation Intentions by Coloring Unicode Strings. In: Asia-Pacific Web Conf. pp. 275–286. Springer (2008)