Using a machine translation tool to counter cross-language plagiarism

HÅKAN ERIKSSON
HAKER@KTH.SE

MARTIN SCHÖN
MSCHO@KTH.SE

Degree Project in Computer Science, DD143X
Supervisor: Christian Smith
Examiner: Örjan Ekeberg

CSC, KTH 29 april 2014
Abstract

Cross-language plagiarism is a type of plagiarism where texts from one language are translated to another, concealing the origin of the text. This study tested if Google’s machine translation tool could be used as a preprocessor to a plagiarism detection tool in order to increase detection of cross-language plagiarism. It achieved positive results for detecting plagiarism in machine translated texts, while craftier plagiarised translations with higher degrees of obfuscation were harder to detect. Utilising even simple tools such as Google’s machine translation tool thus seems to indicate that steps to solving the problem of cross-language plagiarism can be taken.
Chapter 1

Introduction

Plagiarism is often considered a serious and difficult problem to discover within many fields of research, especially if the person committing it is crafty. While a verbatim copy of a text can be discovered with most detection tools given a large enough database of sources, alterations or cross-language translations can often disguise plagiarism. Having effective and accurate tools for countering plagiarism is thus of high importance to maintain the quality of work in fields such as academic study and research.

1.1 Problem statement

This study will attempt to determine if Google’s machine translation tool can be used as a preprocessor for plagiarism detection tools to improve detection of cross-language plagiarism.

The study will only attempt to check the thesis’ validity in plagiarism of academic texts at the bachelor level, not plagiarism of source code or other forms of plagiarism. Creating or modifying the chosen detection algorithm also lies beyond the scope of this study. Instead the aim is to determine if machine translation tools can be used to obtain better results for matching plagiarised texts when utilising an already constructed detection tool that implements a specific algorithm. We will focus on testing a tool that uses a type of substring matching algorithm, this tool is called the Software Similarity Tester (SIM)\(^1\) which will be discussed in section 2.1.

1.2 Background

In this section we will explain necessary background information to this study, such as definitions of plagiarism and the prevalence of plagiarism in academic study. Furthermore, a brief state-of-the-art analysis is discussed.

\(^1\)[http://www.dickgrune.com/Programs/similarity_tester/][Online; accessed 29-April-2014]
1.2.1 Definition

Plagiarism is not a precise term, and the definition is not entirely clear-cut. For the purposes of this study, we will work with the following definition:

- “The act of using another person’s words or ideas without giving credit to that person”

Differing interpretations of what constitutes plagiarism complicates the issue of defining the term unambiguously. For example, given the definition above, “giving credit” is not well-defined. Such phrasing opens up for arbitrariness and confusion for parties affected by plagiarism. Furthermore, different fields of academics have varying requirements on how and what level of citation is needed. This in turn opens up for arbitrariness and confusion for parties affected by plagiarism. Opinions can for example differ between students and faculty, leading to situations where students are not even aware of committing plagiarism [7]. Another dimension to this is that of cultural disparities. Rephrasing words of others can be seen as disguising the use of a source, or that the original text explains the information better than any rewording could [1, 8]. Clearly, the ambiguous definition of plagiarism is something that compounds the problem. However, since this study will focus solely on explicit plagiarism in the form of directly copying excerpts, this particular issue will not affect the study.

Cross-language plagiarism is the specific type of plagiarism that entails taking a source in one language and then translating it to another [6, 9]. This method of obfuscation can often disguise plagiarism since far from all articles or texts are published or stored in other languages than the original. A text that is copied from one language to another can therefore circumvent even advanced detection algorithms, given that few databases include a translated version or some other means to check for cross-language plagiarism.

1.2.2 Prevalence

Measuring the extent of plagiarism in higher education and research is a difficult task for several reasons. Common methods for determining the scale of plagiarism are self surveys and empirical samples. Both approaches have their assorted weaknesses. For one, the question of what constitutes plagiarism and what does not needs to be clarified in any study. As explained in the previous subsection, the definition of plagiarism is open to interpretation which might create arbitrariness to any research about plagiarism.

Self surveys are often especially problematic in this regard since they rely on several facts about the participants. For one, the definition of plagiarism might not align

---

between the researcher and participants, which makes the collected data less reliable. But the most glaring problem with self surveys is that they rely on the participants being honest about behavior that itself is considered dishonest [15].

In theory, directly measuring the occurrences of plagiarism in university settings circumvents the issue of requiring honesty from survey participants. However, this approach also falls short on a few points. For one, it relies on the researcher to actually detect all instances of plagiarism without also adding false positives. Presently there does not appear to exist any tool or method which is perfect in detecting instances of plagiarism [15]. Therefore detection tools and researchers have their limitations in detecting plagiarism, and the problem with definitions and arbitrariness also apply in these types studies.

In conclusion it seems that concrete data on the problem’s frequency is still lacking or at least not as well documented as it could be. Few methods of collecting data seems to yield satisfactory and reliable information. However, they do clearly indicate that the problem of plagiarism exists, with no complete or easy solution at hand, thus showing the importance of studies such as this.

1.2.3 State-of-the-art

Many detection tools available today appears to lack direct means to counter the problem of cross-language plagiarism [2 5]. Researchwise, there are a few different approaches to counter cross-language plagiarism although the attention to this area seems somewhat overshadowed by research into mono-lingual plagiarism detection [6]. In 2013, Barrón-Cedeño et al. published a report that illustrated an overall process to counter cross-language plagiarism and made the source code of the software public [3]. The process consisted of heuristic retrieval, detailed analysis and post-processing. This kind of process is considered to be one of the main approaches for making it possible to scale the plagiarism detection over large document collections [6]. Barrón-Cedeño et al. also evaluated three different models on how to estimate the similarity between two documents when countering cross-language plagiarism [3].

The tool mentioned above utilises external plagiarism detection, which means that it compares potentially plagiarised documents with an collection of legit documents. The cross-language plagiarism problem could also be approached by using algorithms which uses intrinsic plagiarism detection. This method checks potentially plagiarised documents for suspicious changes in the writing style [6]. However, when using the intrinsic approach it could be hard to prove that plagiarism really occurred since there is no source document to act as evidence [13]. Tschuggnall et al. published a report in 2013 with an approach to intrinsically detect plagiarism. In their work they used a corpus from an international competition concerning pla-
giarism detection tools\textsuperscript{3} which includes cases of data written in either Spanish and German and then obfuscated by translating it to English. Their approach showed some promising results compared to other intrinsic approaches. However, they did not present how well the algorithm worked in the cross-language plagiarism cases\textsuperscript{14}.

Every year since 2009 an international competition on plagiarism detection has been arranged, as mentioned above\textsuperscript{3}. During the first three years of the competition they developed a framework for testing different plagiarism detection algorithms. In the beginning of this development the test data consisted of auto generated data from different sources\textsuperscript{11, 12}. Since this data could contain text from sources with different topics, it could not accurately simulate lifelike situations. This property could be exploited by the contestants with an algorithm that checked for topic changes. Today, the framework’s corpus tries to recreate lifelike situation where plagiarism can occur. All the data is manually searched and extracted so that all of the data is about the same topic. Next, the data is transformed into different potential plagiarism cases with different obfuscation techniques\textsuperscript{11}. Before 2012, some test cases consisted of cross-language plagiarism. A few of the contestants had quite successful results against these cases. However these results were considered biased because the test data was obfuscated by the same tool which the contestants used in their plagiarism detection algorithm\textsuperscript{10}. Some promising results were presented in 2012, when this specific problem was evaluated specifically. As of 2013, the competition has not included any tests for cross-language plagiarism\textsuperscript{10, 11}.

On the commercial market, Turnitin, a company that provides a widely used “[...] cloud-based service for originality checking, online grading and peer review [...]”\textsuperscript{4} released a beta feature in 2012 that allowed detection of cross-language plagiarism\textsuperscript{5}. The released feature was said to support 15 different languages and the uploaded content would then be translated into English. After that the translated text would be checked for plagiarism by comparing it to their databases. We could not acquire any more information about this beta and how well it works in practice. To the best of our knowledge the commercial market lack complete solutions to check for cross-language plagiarism as of this study.

\textsuperscript{5}http://pages.turnitin.com/rs/iparadigms/images/Turnitin_RELEASE_TranslatedMatching_ENGLISH.pdf [Online; accessed 29-April-2014]
Chapter 2

Methodology

To answer the question if Google’s machine translation tool is effective as a preprocessor to plagiarism detection tools we conducted a series of tests on the implementation of substring matching explained below. This chapter of the report will first give a general outline of how SIM functions, and then an explanation on how the tests in this study were set up and executed. The last section will explain the motivations behind the chosen methodology and also discuss limitations to the tests.

2.1 The Software Similarity Tester

The plagiarism detection tool that was used in this study is called the Software Similarity Tester and was created by Dick Grune [4], a Dutch computer scientist who is now a retired lecturer at Vrije University in Amsterdam. SIM has been used to detect plagiarism in submitted programs by students attending Computer science workshops at the Vrije University. The program has continuously been updated and supports input file formats of the types C, Java, Pascal, Modula-2, Lisp, Miranda and text files.

Since this study has its focus on academic texts, only the text version of SIM was of interest. The fundamentals of our procedure which incorporates SIM’s algorithm can be described as follows:

1. Read from every text file in the database and in the plagiarised candidates.

---

3. SIM README [http://www.dickgrune.com/Programs/similarity_tester/sim_2_77.zip](http://www.dickgrune.com/Programs/similarity_tester/sim_2_77.zip) [Online; accessed 29-April-2014]
2. Reduce each individual word in every text into a 16-bit hash code. Every text file will now form a long string containing 16-bit characters. Each text in the string is delimited by a special separator.

a. Finds the longest common substring between a candidate and a database text.

b. This substring is then removed from the candidate.

c. Repeat procedure from 2.a. at the next token position subsequent to the earlier found substring until the lengths of these detected substrings decreases to a specified threshold.

3. Calculates the similarity percentages according to the size of the sets of matching substrings between candidates and specific database texts.

4. Prints out similarity percentages between every candidate and its matching database text if there are any.

When a word in a text is tokenized and reduced into a hash code it will only represent letters and/or numbers. Preceding and trailing non-letter and/or number will not be accounted for in the tokenizer procedure and is thus removed from the exhaustive search for matching non-overlapping substrings. The tokens are tokenized by a lex-generated scanner which is modified to be suited for natural language as input⁴.

The minimum length threshold for a matching substring is set to an eight token sequence by default in SIM text². Setting this threshold to a higher value could increase the detection accuracy by constraining the substring matching algorithm to only match on longer strings. Since this algorithm only find matches if the two common substrings are verbatim, a higher threshold would make the algorithm weaker against obfuscated plagiarism. But setting this threshold too low would increase the risk of getting false positives [6]. Therefore the threshold has been left unaltered for the tests in this study.

The similarity percentage is calculated by taking the sum of all tokens in all matching substrings between a candidate and its matching database text. This sum is then divided by the total sum of tokens in the candidate text and then multiplied with one hundred⁵. Note that the sequences of matching tokens consists of a minimum of eight tokens. For a more detailed explanation of the program see the technical report³.

³SIM technical report [http://www.dickgrune.com/Programs/similarity_tester/TechnReport][Online; accessed 29-April-2014]
⁴percentages.c [http://www.dickgrune.com/Programs/similarity_tester/sim_2_77.zip][Online; accessed 29-April-2014]


2.2 Execution

The execution consisted of three parts. First a database was created, then a list of candidates was crafted. These candidates were all plagiarised excerpts from the database, with varying levels of obfuscation and preprocessing. Finally the test was conducted with SIM.

Since the software required a database of original texts, we first added a total of 292 texts to our database. These texts were all taken from bachelor’s thesis’ between 2010-2012 from KTH, specifically from the computer science programme. Since the texts were in pdf-format they were also converted with the tool Pdftotext\(^6\) with the option to maintain physical layout and with UTF-8 encoding. Originally there were 316 texts, but some of them had format conversion issues so we chose to remove them from the database. These 292 texts will reflect the level of plagiarism we wish to study, namely at the bachelor level in university settings. From these texts we then randomly chose 20 texts from the year 2012. The process of randomly selecting the texts was made by using a simple random number generator. From these 20 texts we then selected one or more paragraphs from the “discussion” or “conclusion” portion of the chosen reports. These sections seldom contain advanced technical terms or raw data that might skew the result because of non-existing translations. The chosen excerpts also all fulfilled the following criteria:

- Each paragraph is originally written in English.
- Each paragraph is around 100-150 words long.
- References, if present, are removed.

The criteria were enforced in order to have better control over the test and the subsequent results. We chose to translate from English to Swedish for simplicity’s sake, since translating to a foreign language is more difficult than the opposite. Information and research is also often more extensive and common in English when compared to other languages. The paragraph length constraint was added to ensure that each excerpt is neither too short nor too long when compared to the others. The number itself represents roughly how long an average paragraph is in these texts. Similar to how we wished to limit the amount of technical terms and raw data, we also removed any references such as footnotes or brackets, since these cannot be translated properly. The 20 excerpts that fit these above stated criteria form the basis of our candidates.

Next, we divided the 20 excerpts in five distinct categories. One category was the original excerpts themselves. Two others were the original excerpts translated by hand or by using the machine translation tool. The last two categories represented

---

\(^6\)Portable Document Format (PDF) to text converter \texttt{http://manpages.ubuntu.com/manpages/lucid/man1/pdftotext.1.html} [Online; accessed 29-April-2014]
the preprocessed excerpts. These were the previously mentioned translated plagiarism that had been translated back to the original language using the machine translation tool. Figure 2.1 below demonstrates the entire set-up methodology outlined above:

```
Figure 2.1. Set-up procedure
```

The final portion of the tests was to run the detection tool with the above outlined database and candidates. For the purposes of these tests we also set the detection to the following options:

“[-S] The contents of the new files are compared to the old files only - not between themselves. 
[-p] The output is given in similarity percentages [...] 
[-t N] [...] sets the threshold (in percents) below which similarities will not be reported [...].”

The first option merely ensured that the plagiarised texts were not compared to each other, so that our tests were not cluttered by irrelevant data. The second option is to receive the data in percentages rather than which parts of the texts were matched to the database. Lastly, the default setting in the source code is to ignore matches with lower similarity than 20%. This variable was changed to 1% to show any found matches, no matter how small.

**2.3 Methodology discussion**

The three distinct plagiarism methods outlined above covered basic approaches a plagiarist might take. Since we aimed to study plagiarism at the bachelor level, we were appropriate subjects to make hand translations of the texts since this is the academic level we are currently at. This, combined with direct translation of one or several sources can hopefully emulate examples of actual plagiarism.[11]
Or at least mimic the practice enough to give a hint at if a preprocessor might help detection tools to discover plagiarised texts. However, the ways in which a text can be altered is of course close to infinite. A crafty plagiarist might change the sentence structure, exchange words for synonyms, copy from several sources in several different languages or a combination of them all. This makes the test somewhat less reflective of how real instances of plagiarism might look. The limited set of data is also a source of weakness to the study, a larger test would strengthen the study by lowering the risk of statistical errors.

Another weakness of the tests was the decision to use the machine translation tool as both the plagiarist’s tool and also as the preprocessor for the detection tool. One might imagine that translations back and forth in the same tool give quite similar results, which in turn could skew the results of the tests. However, this slight advantage could be somewhat justified, given that the use of Google’s machine translation tool is quite widespread since it reportedly has over 200 million users worldwide\(^7\).

The used implementation of substring matching in this test can also be challenged. For one, the tool is not currently used in higher education as far as we are aware. Professionally used tools are more likely to receive continuous support and modification, while programs such as the one used in the tests often become obsolete with time. However, professional tools are for obvious reasons seldom publicly available. The underlying algorithms and methods for detection is not always revealed, making tests on these services less transparent and more up for speculation if a preprocessor would be useful. A professional tool might for example already have some sort of translation preprocessor built in, which might cause the tool to be less effective if another one is added to it.

Parameters of the tool itself can, as explained in section 2.1, also be modified. For example, lowering the minimum threshold for a match would likely increase the detection of obfuscated plagiarisms, but also be more prone to create false positives. Arguments can be made both for increasing and decreasing this parameter, depending on the situation and context.

All in all, these issues make studies such as this one less reliable as hard data. Yet the tests can still give indications on whether using a machine translator as a preprocessor can be of some benefit to detection tools.

\(^7\)Google machine translation tool \(\text{https://developers.google.com/international/translation-tools}\) [Online; accessed 29-April-2014]
Chapter 3

Results

In this chapter we will detail the results from the tests conducted according to the methodology specified in the previous chapter. The data is presented and explained in tables and a diagram with corresponding text.

3.1 Detection rates

Table 3.1 provides an overview of the results from all the individual excerpts that were tested. The rows are the detection rates for each report. The values in the table are detection percentages. The method and algorithm for calculating these values are explained in section 2.1. Each column in the table corresponds to our different plagiarism methods, which in short can be described as follows:

- **Original.** These are excerpts from the database in verbatim form.
- **Manual.** These excerpts have been translated by hand from the original language (English) to a foreign language (Swedish).
- **Machine.** These excerpts have been translated through the machine translation tool from the original language (English) to a foreign language (Swedish).
- **Prep manual.** These are the manually translated excerpts but preprocessed back to the original language in the machine translation tool.
- **Prep machine.** These are the machine translated excerpts but preprocessed back to the original language in the machine translation tool.
Table 3.1. Table of detection percentages

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>74</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>95</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>100</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>36</td>
<td>80</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>56</td>
</tr>
<tr>
<td>7</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>37</td>
</tr>
<tr>
<td>8</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>18</td>
<td>37</td>
</tr>
<tr>
<td>9</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>59</td>
<td>63</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>33</td>
<td>32</td>
</tr>
<tr>
<td>12</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>100</td>
</tr>
<tr>
<td>13</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>51</td>
</tr>
<tr>
<td>14</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>41</td>
<td>71</td>
</tr>
<tr>
<td>15</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>96</td>
<td>100</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>32</td>
<td>50</td>
</tr>
<tr>
<td>17</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>33</td>
<td>88</td>
</tr>
<tr>
<td>18</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>19</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>70</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>57</td>
</tr>
</tbody>
</table>

3.2 Statistics

Table 3.2 contains statistical values and properties based on table 3.1. The true mean value interval represents the average match percentages for preprocessed excerpts that are translated manually and through the machine translation tool. The value was calculated by using a two-tailed confidence interval with t-distribution. The data was calculated at 95% confidence.

Table 3.2. Table of statistical analysis

<table>
<thead>
<tr>
<th>Property</th>
<th>Preprocessed manual</th>
<th>Preprocessed machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observed mean value</td>
<td>39.2</td>
<td>76.1</td>
</tr>
<tr>
<td>Observed standard deviation</td>
<td>31.82</td>
<td>25.73</td>
</tr>
<tr>
<td>True mean value interval</td>
<td>39.2 ± 15.26</td>
<td>76.1 ± 12.34</td>
</tr>
</tbody>
</table>
3.3 Comparison of detection rates

Figure 3.1 shows the detection percentages from table 3.1 for preprocessed excerpts that have been translated either manually or by the machine translation tool. The X-axis corresponds to the excerpts’ report number from table 3.1. The Y-axis represents the detection percentages as explained in section 2.1. The solid line is the machine translated excerpts, while the dashed line represents the manually translated excerpts.

![Detection rates for preprocessed excerpts](image)

**Figure 3.1.** Detection rates for preprocessed excerpts
Chapter 4

Discussion

In this chapter we will discuss the results obtained from the tests performed in this study, as well as observations concerning the general problem of cross-language plagiarism. Comparisons between the tests as well as implications of them will be analysed. Finally, a few suggestions for improving or expanding on these tests are addressed.

As mentioned in [1.2.3], there are several methods to deal with the problem of cross-language plagiarism, but far from all are reliable or efficient enough in practice. For example, a solution could be to add translated copies of documents into plagiarism databases but is likely a difficult and impractical solution, for several reasons. The work needed to properly translate every article or text in one language to another would be huge, although the problem can be alleviated by using machine translation tools. But even if one would settle for raw machine translations, the articles would require large amounts of storage capacity. Each new supported language would double the original database size. The computation time needed to check a work for plagiarism in this database would therefore also increase together with the size of the database.

A more feasible solution would of course be to translate only the suspected text, which is what was tested in this study. This solution would still increase computation time needed to match texts with the database together with the number of translations. The problem of translating the text would also remain largely the same; a direct machine translation would of course mostly catch careless plagiarists, but trying to “predict” custom types of translations would be a problem in and of itself. The upside of this solution would of course be that the databases’ size would remain unaffected. As we have seen it is also relatively easy to implement. Before starting the detection tool one would simply have to parse the suspected text through a translation tool.

The used detection tool seems to function well on verbatim plagiarism. As expected,
all 20 of the excerpts were matched fully to the database when they were parsed without any alteration (table 3.1). However, the detection tool could not detect cross-language plagiarism at all without the aid of the machine translation tool. Plagiarised documents that were not preprocessed all passed the screening without detection. This is no surprise, seeing as how the tool has no native function to test for this kind of plagiarism. Substring matching as an algorithm can only compare texts in the same language, rendering it very vulnerable to cross-language plagiarism.

This piece of data highlights a problem of many detection tools available today. Far from all detection tools have taken measures to deal with this kind of plagiarism, as mentioned in section 1.2.3. How these measures could look depends greatly on the type of underlying algorithms and heuristics of individual detection tools. Preprocessors may be viable for certain detection tools while other algorithms may benefit from other solutions. Either way, it is apparent that a dedicated effort is needed to counter cross-language plagiarism, and preprocessors such as the one used in these tests certainly seem to be an option.

When the machine translation tool was utilised as a preprocessor, detection rates improved greatly. Plagiarism conducted through machine translations had a very high detection rate as we have seen (figure 3.1). However, this can likely be attributed to the fact that the machine translation tool was used both as the plagiarism tool as well as preprocessing tool. As mentioned in section 2.3 it can be assumed that translations back and forth skewed the results in its favour. Even so, the results show that a careless plagiarist can be caught even with these simple measures.

The manual translations were on the other hand more difficult to detect. The results are sharply distinct from that of the machine translations (figure 3.1). This difference in results between the two plagiarism methods can be attributed to several factors. For one, the machine translations can, as mentioned earlier, take advantage of the fact that it is used both as a plagiarism tool as well as a preprocessor. Another difference is that manual translations are almost always obfuscated when compared to machine translations. Machine translations are literal translations, they can seldom translate a text without causing some error or mismatch in a text’s context or meaning. A manual translator on the other hand has every incentive to try to distance their plagiarism from the original text as well as trying to correct errors that machine translations often commit. An example of this is that we chose to remove references and footnotes if they were present, a decision that could have had a negative impact on the detection rates. Therefore one can safely assume that obfuscation that arises from more advanced translations causes the detection rates to drop. Crafty plagiarists may in this way avoid being caught by ensuring that their translation is obfuscated.

Several improvements can be made to expand on these tests. An obvious method is
to execute the tests on more plagiarism detection tools. This might give insights into how effective the use of machine translation tools as a preprocessor are depending on the underlying algorithms. The tests in this study only involved an implementation of the substring matching algorithm but as we have seen in section 1.2.3, there is a wide array of approaches to detecting plagiarism in currently available software. One can therefore not assume that this test is representative for all algorithms, it may work better or worse in other detection strategies.

Statistically speaking, the test is not entirely conclusive either (table 3.2). The data sample is small, decreasing the reliability of our data. Nor can the variations of translations used in this study be said to accurately represent all kinds of manual cross-language plagiarism. The number of ways to translate a text are close to endless, depending on the plagiarist’s skill and the tools’ he or she has available. Therefore, more tests on translations with different criteria and levels of obfuscation would strengthen the reliability of our results. It would also further define the strengths and weaknesses of the detection algorithm depending on the level of obfuscation and other properties of the plagiarism.
Chapter 5

Conclusion

The tests conducted in this study indicate that Google’s machine translation tool is a viable approach to countering cross-language plagiarism for some plagiarism detection tools. Our results show that the tested plagiarism detection tool’s underlying substring matching algorithm had improved detection rates when the plagiarised excerpts were preprocessed. Machine translated examples of plagiarism showed promising results of being detected, while more obfuscated manual translations were more difficult. It can therefore be argued that detection tools using implementations of the substring matching algorithm can benefit from Google’s machine translation tool as a preprocessor. The data is however less conclusive on detection tools utilising other algorithms. Other algorithms may handle obfuscation such as translations differently, which makes the results of this study less general. However, the data indicates that a dedicated solution for cross-language plagiarism is needed for all detection tools, but whether or not machine translation tools is an appropriate measure remains to be researched further.


