A Study on Plagiarism Detection and Plagiarism Direction Identification Using Natural Language Processing Techniques

Man Yan Miranda Chong

A thesis submitted in partial fulfilment of the requirements of the University of Wolverhampton for the degree of Doctor of Philosophy

2013

This work or any part thereof has not previously been presented in any form to the University or to any other body whether for the purposes of as- sessment, publication or for any other purpose (unless otherwise indicated). Save for any express acknowledgements, references and/or bibliographies cited in the work, I confirm that the intellectual content of the work is the result of my own efforts and of no other person.

Signature: . . . .

Date: . . . .

This thesis is dedicated to the memory of my grandfather, Yuk Ming Chong.

“Iron is full of impurities that weaken it; through the forging fire, it becomes steel and is transformed into a razor-sharp sword. Human beings develop in the same fashion.”

Morihei Ueshiba

Abstract

Ever since we entered the digital communication era, the ease of information sharing through the internet has encouraged online literature searching. With this comes the potential risk of a rise in academic misconduct and intellectual property theft. As concerns over plagiarism grow, more attention has been directed towards automatic plagiarism detection. This is a computational approach which assists humans in judging whether pieces of texts are plagiarised. However, most existing plagiarism detection approaches are limited to superficial, brute-force string-matching techniques. If the text has undergone substantial semantic and syntactic changes, string-matching approaches do not perform well. In order to identify such changes, linguistic techniques which are able to perform a deeper analysis of the text are needed. To date, very limited research has been conducted on the topic of utilising linguistic techniques in plagiarism detection.

This thesis provides novel perspectives on plagiarism detection and plagiarism direction identification tasks. The hypothesis is that original texts and rewritten texts exhibit significant but measurable differences, and that these differences can be captured through statistical and linguistic indicators. To investigate this hypothesis, four main research objectives are defined.

First, a novel framework for plagiarism detection is proposed. It involves the use of Natural Language Processing techniques, rather than only relying on the traditional string-matching approaches. The objective is to investigate and evaluate the influence of text pre-processing, and statistical, shallow and deep linguistic techniques using a corpus-based approach. This is achieved by evaluating the techniques in two main experimental settings.

Second, the role of machine learning in this novel framework is investigated. The objective is to determine whether the application of machine learning in the plagiarism detection task is helpful. This is achieved by comparing a threshold-setting approach against a supervised machine learning classifier.

Third, the prospect of applying the proposed framework in a large-scale scenario is explored. The objective is to investigate the scalability of the proposed framework and algorithms. This is achieved by experimenting with a large-scale corpus in three stages. The first two stages are based on longer text lengths and the final stage is based on segments of texts.

Finally, the plagiarism direction identification problem is explored as supervised machine learning classification and ranking tasks. Statistical and linguistic features are investigated individually or in various combinations. The objective is to introduce a new perspective on the traditional brute-force pair-wise comparison of texts. Instead of comparing original texts against rewritten texts, features are drawn based on traits of texts to build a pattern for original and rewritten texts. Thus, the classification or ranking task is to fit a piece of text into a pattern.

The framework is tested by empirical experiments, and the results from initial experiments show that deep linguistic analysis contributes to solving the problems we address in this thesis. Further experiments show that combining shallow and deep techniques helps improve the classification of plagiarised texts by reducing the number of false negatives. In addition, the experiment on plagiarism direction detection shows that rewritten texts can be identified by statistical and linguistic traits. The conclusions of this study offer ideas for further research directions and potential applications to tackle the challenges that lie ahead in detecting text reuse.

Acknowledgements

There are many people who I would like to thank, but if I list all of them here then this section would probably match the entire length of the thesis. I have to keep this brief and I hope I have not missed anyone of importance.

First and foremost, I would like to thank my director of study, Prof. Ruslan Mitkov, for giving me the very privileged opportunity to work under his tutelage. His guidance has made the challenging PhD achievable. I would also like to thank my supervisor, Dr. Lucia Specia, for her continued support and advice which always work wonders. Without her help this PhD would be an impossible task.

Special thanks go to the colleagues from the Research Group in Computational Linguistics, especially Alison Carminke and Erin Stokes who proofread the early drafts, and Emma Franklin and Stephanie Kyle who proofread the more matured drafts. I would like to thank Grant Dyer, Laura Hasler and Carmen de Vos Martin who have gone out of their ways to proofread my thesis.

I would also thank the lecturers and staffs from the University of Wolverhampton, especially Mr. Peter Wilson and Mr. Tony Proctor, for their insightful advice and support during my undergraduate studies, which led to this PhD.

A few years back I would not have the wildest imagination that I could accomplish a PhD. After overcoming some adversities in Hong Kong I found myself in England pursuing higher education. Settling for a new life proved to be an exciting challenge, which would have been impossible to overcome without the upbringing and education that I received in my early years. I would like to thank my family and friends, especially my parents Dr. LC Chong and Ada Chong whom I cannot thank enough, for their unconditional love and support all these years, who always see the best in me and believe in everything I do. I would also like to thank the sisters, teachers and schoolmates from the Yaumatei Catholic Primary School (AM) and the Tak Nga Secondary School who taught me to be the best that I can be, and most importantly, to be a person who embraces traditional values, typified by the virtues benevolence, rectitude, knowledge, courage, diligence, perseverance, respect, honesty and loyalty.

I would like to extend my thanks to the instructors and members of the Wolverhampton Uni Aikido Club and the Aikido Fellowship of Great Britain for their supplement of stress relief training sessions, which were totally needed during the PhD. I would also like to thank my instructor, Sensei Vince Leadbeatter, for showing me the importance of patience and dedication in and out of the dojo.

Finally, I would describe my experience of the PhD study as a marathon. There were people who helped me to prepare at the start, there were people along the way to cheer me on, and there were people at the finish waiting for me to cross the line. Those who were along the way may be seen briefly and then gone, but of course their encouragement would not be forgotten. Without their support the journey would have been far too much to bear on my own. To the few people who were my support team from start to finish, I cannot thank you enough - my gratitude is beyond words.

Contents

List of Tables

List of Figures

List of Abbreviations

ED	Edit Distance
K-LSD	Kullback-Leibler Symmetric Distance
LCS	Longest Common Subsequence
LM	Language Model
LSA/LSI	Latent Semantic Analysis/Indexing
NLTK	Natural Language Processing Toolkit
NWV	Normalised Word Vectors
POS	Part-of-Speech
RIPPER	Repeated Incremental Pruning to Produce Error Reduction
RKR-GST	Running-Karp-Rabin Greedy-String-Tiling
RTE	Recognising Textual Entailment
STM	Statistical Machine Translation
STS	Semantic Textual Similarities
SVD	Singular Value Decomposition
SVM	Support Vector Machines
tf-idf	Term frequency-inverse document frequency
TK	Tree Kernel
VSM	Vector Space Model
WSD	Word Sense Disambiguation

Chapter 1

Introduction

Before the era of the World Wide Web, searching for information used to take an enormous amount of time and resources, digging through paper archives and books. Nowadays, information is easily accessible for the internet-enabled generation without having to lift a finger - or literally, at the click of a finger. There are, however, disadvantages notwithstanding the ease of access. Plagiarism poses an increasing challenge to society, which affects academia and the publication industries in particular. In an attempt to maintain academic integrity and protect intellectual property, educational institutions and publishing houses have resorted to the use of plagiarism detection services. However, these commercial tools are very limited and it is complicated to deal with cases in which the ownership of the original source text is disputable.

1.1 Plagiarism

Plagiarism, which is the act of passing off somebody else's original words and ideas as one's own, is seen as a moral offence and often also a legal offence. Plagiarism has an ancient root, as the word itself is derived from the Latin words "plagiaries", which means abductor, and "plagiare", which means to steal. The dictionary definition of plagiarism is “The action or practice of taking someone else’s work, idea, etc., and passing it off as one’s own; literary theft.” (Oxford English Dictionary[1]). Plagiarism has become a major concern since the establishment of education assessment. Since we entered the internet era, the fast, vast, and easy access of information has further escalated the problem of plagiarism.

Plagiarism exists in many different scenarios, and is often difficult to prove or solve. From a modern educational perspective, the rise of the internet as an information sharing platform has provided students with more ways to access electronic materials. At the same time, essay banks and ghost writing services known as “Paper Mills” appeared. According to an internet survey by the Coastal Carolina University[2], the list of Paper Mills in the US has soared from 35 in 1999 to over 250 in 2006, and to date the figure is still rising.

Contrary to popular belief, students are not the only ones who face scrutiny. Apart from academic misconduct charges, plagiarism can also cause financial and reputation losses. There have been a number of scandals where high-profile authors were caught plagiarising in the publication industry, and others where even government ministers were caught plagiarising their PhD theses. There have also been cases where academics reused large parts of text for funding proposals. For instance, a case study[3] which surfaced in May 2010 revealed that a book written by a professor of mathematics and algebra was in fact a plagiarised work from another professor. The original book was written in German and the plagiarised

[1] http://www.oed.com/view/Entry/144939

[2] http://www.coastal.edu/library/presentations/mills2.html

[3] http://www.zeit.de/studium/hochschule/2010-05/mathematik-plagiate

version was translated word-for-word, possibly by using machine translation tool, and published in English by a large publishing house. The publishing house had to withdraw the plagiarised book after the case attracted attention on the Internet. In 2011 it was discovered that the doctoral thesis of the German defence minister consisted of large amount of plagiarised texts.[4] Within a week his doctorate was rescinded, and he stepped down from his role. Needless to say, these examples are only the tip of the iceberg. More case studies are listed in Appendix E.3.

As more and more information becomes available online, the sheer amount of information for manual investigation becomes overwhelming. Hence, computational methods have been introduced to aid text reuse, authorship and direction identification. This is where automatic plagiarism detection started to gain attention, as it may be able to offer an effective and efficient solution, at a lower economic cost than using human resources.

1.2 Plagiarism Detection

In the early days, plagiarism could only be detected manually by relying on the readers' own knowledge. As cognition varies from person to person, and the vast amount of materials is impossible to attain, the process of identifying plagiarism within text can be a difficult task. In most cases, plagiarism is identified by reading a text that triggers a "Déjà vu" in the reader, where the reader has recognised it. The obvious disadvantage of the manual method is that when the amount of information increases, a reader is less likely to be able to identify the similarities,

[4] http://www.tagesschau.de/inland/guttenberg762.html

as the human brain does not function like a computer hard-disk where information is easily accessible on demand.

One of the earliest methods of plagiarism detection was introduced by Bird (1927), which investigated the application of statistical methods in detecting plagiarism of multiple-choice answers. Later methods developed through the 1960s were focused on detecting plagiarism in multiple-choice tests. Early plagiarism detection systems for written texts started to appear around the 1990s. These tools used statistical methods to calculate similarity between texts, and most tools focused on written-text plagiarism while some focused only on computer source code plagiarism. A detailed account of early research in plagiarism detection systems is described in Section 3.1.

In the last decade, commercial systems have flourished thanks to the increase in student numbers and assignments. In 2000, there were only five established systems, four of which were used for identifying written-text plagiarism and one for identifying source-code plagiarism (Lathrop and Foss, 2000). A decade on, in 2010, 47 systems were noted (Weber-Wulff, 2010). This substantial growth suggests that plagiarism has not been dealt with effectively, thus many tools have been developed to meet the increase in market demand.

The use of plagiarism detection systems has become the standard practice in many higher education institutions. In the UK, many universities have been advised by the Joint Information Systems Committee (JISC)[5] to adopt the online

[5] http://www.jisc.ac.uk/whatwedo/topics/plagiarism.aspx

service Turnitin©[6]. It provides a similarity check against its own database that contains archives of all previously submitted student papers, and access to web journals and books. The text similarity detection algorithms used in commercial systems are commercial secrets, but simple test cases which contain some level of paraphrasing and structure changes have shown that it is possible to bypass detection.

The inadequacy of existing systems has sparked research into plagiarism detection. There are various approaches of plagiarism detection and they usually comprise three main stages: 1) text pre-processing, 2) filtering and 3) detection. However, existing approaches are mostly limited to exact comparisons between suspicious plagiarised texts and potential source texts at the character or string level. The accuracy of these approaches is yet to reach a satisfactory level and plagiarism continues to affect many areas, especially in the field of education and publishing. A discussion of the advantages and disadvantages of plagiarism detection methods is listed in Section 3.1.

The biggest challenge in the plagiarism detection field is that most approaches are inadequate at detecting texts with substantial semantic and syntactic changes. For a human it is easy to understand texts which carry similar meaning even when they are rewritten using different words and structures. However, computers are unable to understand texts in a similar manner, especially when automatic detection relies on exact text matching. A possible solution to this challenge lies in the research area of computational linguistics, which provides techniques for

[6] http://submit.ac.uk/en_gb/home

aiding deeper linguistic analysis. The use of such techniques is still an under-explored area in the plagiarism detection field. In order to shed light on the existing plagiarism detection approaches, this thesis henceforth proposes the use of linguistic techniques to investigate the deeper meaning of text in plagiarism detection.

1.3 Aims and Objectives

The main aim of this thesis is to investigate the use of Natural Language Processing (NLP) techniques in text reuse detection and direction identification tasks. The hypothesis is that original texts and rewritten texts exhibit significant and measurable differences, and that these can be captured through statistical and linguistic indicators. To test the hypothesis, a framework which incorporates NLP techniques along with existing shallow techniques is proposed to improve the identification of plagiarised texts.

The scope of the research is limited to external plagiarism, where both the suspicious plagiarised texts and the potential source texts are available. All plagiarised texts and source texts are monolingual English written texts. The evaluation is based on an empirical corpus-based approach, where different corpora are used to test various experimental settings. When applicable, supervised machine learning models are used for text classification. Initial experiments refer to plagiarised text cases as “plagiarised documents”, which in later experiments are referred to as “plagiarised passages” as the plagiarised text length changes.

More specifically, this thesis attempts to answer the following research questions:

- How can Natural Language Processing techniques be used to improve the performance of existing approaches?
- Does machine learning bring any benefits to the plagiarism detection framework?
- Will the framework perform well in a small-scale scenario as well as a large-scale scenario?
- Can the task of identifying the direction of plagiarism benefit from the investigation of statistical and linguistic traits?

To answer the above research questions, the following objectives need to be met: The **first objective** is to propose a framework which incorporates NLP techniques and the string-matching approach. The task is to identify text pre-processing, statistical, shallow and deep linguistic techniques which can improve traditional approaches. The influence of the techniques is investigated and evaluated using a corpus-based approach.

The **second objective** is to investigate the role of machine learning in the proposed framework. This is achieved by adopting a supervised machine learning classifier in the framework to support the decision between plagiarised and non-plagiarised cases, or among several levels of plagiarism. This approach is evaluated against non-machine learning approaches.

The **third objective** is to evaluate the scalability of the proposed framework and algorithms. This is achieved by performing experiments in a small-scale scenario and a large-scale scenario, with corpora that contain cases of varied text lengths.

The **final objective** is to explore the identification of plagiarism direction, by proposing a framework that investigates statistical and linguistic traits of texts. The task is to establish a new perspective on the traditional plagiarism detection approach. Instead of comparing many suspicious texts against many source texts, features are drawn based on traits of texts to build a pattern for original and rewritten texts, allowing a text case to be fitted into a pattern rather than using the traditional pair-wise comparison. The decision, in this case, is between whether the text is original or plagiarised, but the indication of the source of plagiarised cases is not a concern.

1.4 Overview of the Framework

1.4.1 Approach for external plagiarism detection

The proposed framework aims to enhance the existing string-matching plagiarism detection approach with NLP techniques. The framework is organised as a five-stage approach. The operation of the stages is dependent on the input data, where in some cases not all the stages are required for specific tasks.

Stage 1: Pre-processing This stage is to prepare the input data, i.e., the entire text collection of suspicious and source texts (corpus), with the language processing techniques which include simple text processing and shallow NLP techniques. This step generalises the data for feature extraction or comparison in other stages.

Stage 2: Similarity comparison This stage is to perform pair-wise comparison for all processed texts using a similarity metric. The similarity between text-pairs is given by a similarity score, which is then passed on to Stage 3.

Stage 3: Filtering The similarity scores generated in Stage 2 are used for judging the likelihood for a text-pair to be listed as a candidate pair. The likelihood is usually determined by setting a threshold on the similarity scores. The text-pairs with higher similarity scores are selected for further processing and the rest are discarded. This reduces the search span in the deep linguistic processing stage.

Stage 4: Further processing Further processing involves the application of deeper language processing techniques, which are computationally expensive to be applied on the whole corpus. When the candidate pairs are retrieved, they are processed by one or more of the modules, generating one or more additional similarity scores.

Stage 5: Classification The final stage is to give each text pair a classification as *Plagiarised* or *Non-plagiarised.* In some cases the *Plagiarised* class can be further defined in various levels, such as *Near Copy*, *Heavy Revision*, or *Light Revision.* The classification is either done by setting thresholds on the scores from Stage 4, or by using similarity scores generated from various modules in that stage as features in a supervised machine learning classifier.

1.4.2 Approach for plagiarism direction identification

The proposed framework aims to bring a novel perspective to the traditional pairwise comparison detection approach. The framework is organised as a three-stage approach. As opposed to the traditional external plagiarism detection approach where *plagiarised cases* and *source cases* are treated as a pair, the identification of plagiarism direction requires each *plagiarised case* or *source case* to be treated on their own. This is done by drawing statistical and linguistic features from each case that can represent rewriting or originality traits. Such features are evaluated individually and in various combinations as supervised machine learning classification or ranking tasks. This sheds light on a number of potential applications, such as first-stage filtering in the traditional plagiarism detection approach, or intrinsic plagiarism detection and authorship identification. The framework has three main stages:

Stage 1: Pre-processing This stage is to prepare both plagiarised text segments and original text segments with language processing techniques, which include simple text processing, shallow and deep NLP techniques. This stage generalises the input data for subsequent stages.

Stage 2: Feature extraction Morphological, syntactic and statistical traits are extracted and used as individual feature sets or combined feature sets.

Stage 3: Classification The final stage is to classify or rank each case into its respective class. This can be a binary classification task to classify each case as *Plagiarised* or *Original*, or a ranking task to sort a plagiarised and original pair according to which version is the most original.

1.5 Structure of the Thesis

The remainder of this thesis is organised in two main parts:

Part 1, which is covered in Chapters 2 and 3, contains the definition of plagiarism and plagiarism detection, and related work.

Part 2, which is covered in Chapters 4-7, contains the in-depth description of the original contributions of this thesis.

Chapter 2 defines the important concepts related to plagiarism. The chapter gives a clear definition of what constitutes plagiarism in an experimental setting, which is used throughout the thesis. It also introduces various types and characteristics of plagiarism. Furthermore, it lists the information used in automatic plagiarism detection methodologies and the main types of methodologies. The chapter concludes with a general description of evaluation approaches used in automatic plagiarism detection.

Chapter 3 covers existing plagiarism detection and direction methodologies, including early approaches and state-of-the-art approaches. The chapter also describes the role of NLP in plagiarism detection and direction identification, the limitations of existing approaches, and other related work.

Chapter 4 provides a detailed description of how NLP techniques are applied in our plagiarism detection framework. It first outlines a general framework which is used throughout the thesis, then describes the text pre-processing and NLP techniques used in the experiments listed in Chapters 5 and 6. The rest of the chapter describes the similarity metrics, machine learning algorithms and evaluation metrics used.

Chapter 5 describes the experiment performed on a small-scale corpus. It covers some general information about the corpus used, followed by the text pre-processing and NLP techniques applied to the corpus, and the similarity metrics and evaluation metrics used.

Chapter 6 describes the experiment performed on a large-scale corpus, on two distinct processing levels: document level and passage level. The chapter covers the information about the corpus used, followed by how the corpus is prepared for the document-level and passage-level experiments. It then describes the text pre-processing and NLP techniques applied to the corpus, the similarity metrics and finally the evaluation methods used.

Chapter 7 describes the experiment on the identification of plagiarism direction performed on segments of texts with various plagiarism levels. The chapter covers the corpus used and the proposed framework, followed by the text pre-processing and NLP techniques applied to the corpus, then by a list of similarity metrics and machine learning algorithms applied in the experiment. The theoretical motivations are described in the feature extraction and selection section, followed by the evaluation and results of the experiment.

The final chapter, Chapter 8, sums up the thesis by recapitulating its objectives, offering a critical evaluation of how successfully these objectives were addressed, and finishes by suggesting a few further research directions.

Chapter 2

Preliminary Notions

This chapter introduces the concepts and terminology which are necessary to understand the topic of this research. The chapter starts off with the definition of plagiarism in a research context, then presents the different types of plagiarism and the characteristics of plagiarism. The notion of automatic plagiarism detection and types of plagiarism detection methodologies are explained, along with a description of the general pipeline of plagiarism detection and evaluation. The chapter concludes with the notion of plagiarism direction detection.

2.1 Definition of Plagiarism

According to the Oxford English Dictionary definition, plagiarism is:

> "*The action or practice of taking someone else's work, idea, etc., and passing it off as one's own; literary theft. (Oxford English Dictionary*[7]*)*"

Plagiarism is not considered to be a black-and-white issue, as there remain many grey areas. Studies have stated that the concept of plagiarism is vague and it is very difficult to give a fixed definition (Piao et al., 2001; Brin et al., 1995; Clough, 2003).

[7] `http://www.oed.com/view/Entry/144939`

In modern terms, the definition of plagiarism is largely influenced by human subjectivity and it is sometimes blurred with other issues, such as intellectual property theft, copyright infringement, and text re-use in domains such as journalism. In some cases, reusing one's own materials is regarded as copyright infringement, which is also known as self-plagiarism. Another form of plagiarism also considers collusion, where two pieces of work appear similar as two authors work together, despite the requirement demanding individual work (Badge and Scott, 2009).

A technical definition of plagiarism is given by Sorokina et al. (2006), where plagiarism is defined as a sequence of word n-grams from one document that appears in another document as consecutive words, or the same sequence of words substituted by their synonyms. However, this definition does not cover cases where orders of words are changed, or paraphrasing that consists of changes as in active/-passive voice.

This brings the need to answer the question of "what attributes make a real plagiarism case?" for this thesis. In our research context, with the goal of proposing approaches to detect plagiarism, we define a plagiarism case as follows:

- A plagiarism has a sequence of words, also known as word n-grams, which have been either directly copied or paraphrased from one source to another.
- A plagiarism case can be of various lengths; plagiarism can exist in an entire document, or within segments of a document.
- A plagiarism case is a segment that is annotated in a corpus usually artificially created for empirical research purposes, instead of containing dis-

putable real-life plagiarism cases.

As the focus of this thesis is not to define and justify the frontier between plagiarism and the other aforementioned issues, the above definitions should be sufficient for setting the specification of the experiments described in subsequent chapters.

2.2 Types of Plagiarism

Plagiarism comes in many forms. It can happen in any field that involves a creation process, which includes written text, computer source code, art and design, and even music pieces. As the focus of this thesis is on written text only, the details of other types of plagiarism will only be briefly mentioned.

The types of plagiarism which have been addressed in previous research are mainly:

- Multiple-choice tests.
- Source code in programming languages.
- Written text, also known as free text and natural language text plagiarism.

Plagiarism in multiple-choice tests and source code is very different from plagiarism in written text. Detecting plagiarism in multiple-choice tests relies on statistical approaches in which the number of matching incorrect answers between two tests is compared to the normal distribution of similar incorrect answers in the collection. On the other hand, source code plagiarism detection requires different tools and metrics which captures statistical features to determine similarities between the codes. In this study, the focus is on written text as it poses a greater challenge, and linguistics features can be investigated alongside statistical features.

For written text plagiarism, the most common cases are found in academic settings. Educational institutions commonly have a set of rules that list what is considered to be plagiarism. The following are examples of how plagiarism can occur in academia (Maurer et al., 2006):

- Ghost writer/submitting someone else's work
- Insufficient referencing
- Direct copying, from one or multiple sources
- Paraphrasing

The above cases can occur in two types of text:

- Monolingual (copied from one language)
- Cross-lingual (copied from a second language, sometimes known as translated plagiarism)

To keep within the scope of the study, in the rest of this thesis, the term "plagiarism" refers to cases where monolingual English written texts have been copied directly or paraphrased from one or more original sources.

2.3 Characteristics of Plagiarism

The characteristics of plagiarism are often observable from statistical and linguistic traits. There are several factors that can indicate a plagiarism case (Clough, 2000), as we discuss in what follows.

2.3.1 Lexical changes

Lexical changes involve the addition, deletion or replacement of words in the text. A sudden change of vocabulary, such as the excessive use of new terminology within a document, is usually a good indication of copy-and-paste plagiarism. Another example is the word-for-word substitution by synonyms. This type of plagiarism is undetectable using the traditional string-matching approach. Detection would require the analysis of lexical information throughout the text.

2.3.2 Syntactic changes

Changes in syntactic information are best observed from significant rearrangement of the structure of the text. Examples include word/clause re-ordering, active versus passive voice, etc. Similarities in syntactic structures can be an indication of plagiarism, but again it is undetectable using the traditional string-matching approach, and detection would require the analysis of syntactical structure of text.

2.3.3 Semantic changes

This involves more radical changes in the text, normally based on heavy paraphrasing that can include both lexical and syntactic changes. Detecting this type of change would require the analysis of semantic information to judge whether two texts hold the same meaning. Again, this is undetectable with the traditional approaches.

2.4 Automatic Plagiarism Detection

This section covers basic notions on automatic plagiarism detection approaches, including plagiarism corpora needed to create and/or evaluate such approaches, the various types of detection and evaluation methodologies.

2.4.1 Corpora of plagiarised texts

First and foremost, existing plagiarism corpora very rarely consist of real plagiarism cases. The reason is that naturally occurring plagiarism cases are hard to obtain. Artificially and simulated plagiarism cases such as the PAN plagiarism detection competition corpus (Potthast et al., 2010b) are needed because the acquisition of real plagiarism cases is often laden with social, legal and ethical issues, along with other technical concerns.

The social concern is that publishing results generated from real data may damage an individual's or an organisation's reputation, which may result in potential lawsuits. The legal and ethical aspect of using real plagiarism cases in a corpus is that it will require the consent of both the original author and the person committing the plagiarism act. Needless to say, it is very rare for someone to actually admit to plagiarism, let alone give consent for the proof of their illicit act to be used as a case study. The ethical concern of making the corpus publicly available is that even if the texts had been made anonymous, in some cases it would still be possible to identify the author. Finally, the technical difficulty of using real plagiarism cases is that the size of the corpus needs to be large enough. In order to facilitate an empirical study, annotations containing the details of the plagiarised texts along with their original texts need to be made available.

These concerns resulted in major difficulties in using real plagiarism cases in the plagiarism detection field. Hence, most research uses artificial plagiarism cases which are either generated by computational methods or at a high cost of manual resources. Following the terminology commonly used in the field, corpora created for the empirical study of plagiarism detection normally contains the following:

Suspicious cases These are suspicious texts that are either non-plagiarised (clean cases) or plagiarised (contains various levels of plagiarism).

Source cases These are the potential original texts that may be partially or entirely copied by the plagiarised cases.

Annotations These are the labelling of each plagiarised case. In some corpora the list can be very comprehensive, for example listing the start and end position of the plagiarised texts, the length of the plagiarised texts, and the same for the associated source texts. Other corpora may have a simpler list, only listing the suspicious-source case pairs at the document level.

In subsequent chapters, we use the term plagiarism corpus to refer to a dataset with cases for modelling plagiarism detection approaches.

2.4.2 Types of plagiarism detection approaches

To define "plagiarism detection" for this thesis, it is first necessary to identify the type of approach for the system. There are two main types of detection approaches, which refer to the detection task given a type of plagiarism corpus.

Intrinsic detection

An "intrinsic" approach refers to cases where plagiarism is to be detected based on a single piece of text, which may contain both non-plagiarised and plagiarised passages. The detection task aims to identify plagiarised passages within that text, without referring to any potentially original text.

Extrinsic/External detection

An "extrinsic", or more commonly, "external" approach refers to cases where sets of suspicious plagiarised texts and their potential original source texts are both available. The detection task aims to identify pairs of matching suspicious-source cases, by analysing the similarity of each suspicious case against a (often very large) collection of potential original cases.

Hybrid detection

A "Hybrid Approach" is the combination intrinsic and external detection. This is more likely to be applied as an improvement to the filtering stage where external detection is used as a filtering strategy, and then intrinsic detection is applied to identify the location of the plagiarised passage, and vice-versa.

For external and hybrid approaches, one can distinguish between an "online approach" and an "offline approach". An "online" approach performs comparisons not only from a local dataset, but also searches the web for texts that may be the original documents. An "offline" approach is based on detection algorithms to identify evidence of plagiarism within a local text collection.

Monolingual detection A "monolingual" detection approach treats the suspicious cases and the source cases in the same language. Suspicious cases are derived from the source cases without any changes to the lanaguage.

Cross-lingual detection A "cross-lingual" detection approach is needed when the suspicious cases are derived from source cases of different languages. The derived texts are then translated by manual or automatic means. This approach typically requires language generalisation as part of the pre-processing stage.

In this thesis, the focus is on *external detection* of monolingual texts in English, within an *offline approach.* In addition, our plagiarism detection approach provides an indication of potentially plagiarised case pairs, instead of identifying exactly which parts of the text have been plagiarised.

2.4.3 General framework for external plagiarism detection

The external plagiarism detection task follows a general framework that involves three main stages of processing. The three stages are: text pre-processing, filtering and detection. This general approach provides the foundation for the detection framework proposed in this thesis and described in Chapter 4.

Generally, external plagiarism detection approaches are achieved through a large number of pair-wise comparisons, by comparing each suspicious case against all source cases in the collection. In order to facilitate further comparison, both the suspicious cases and the source cases can be generalised with text pre-processing techniques such as tokenisation and lowercasing.

After the cases has been processed, pair-wise comparisons between suspicious cases and source cases will begin. Usually this comparison uses superficial word-overlap metrics and a similarity score is generated for each case pair. As this is a brute-force approach, structural metrics based on computationally expensive deeper NLP techniques cannot be applied efficiently. Thus, a "filtering" stage is needed to rule out source cases that do not exhibit significant evidence of being a potential plagiarism source. For example, a similarity metric based on n-gram word overlap between suspicious and source document pair serves well as a filtering strategy. If the similarity score of a case pair is above a certain pre-selected threshold, then the pair will be passed on to the next processing stage. On the other hand, if the pair is below the threshold, it will be excluded from further investigations.

The "detection" stage refers to the classification given to each case pair after filtering and further processing. Classification is done using more advanced and costly similarity metrics with or without the aid of machine learning algorithms, where one or more similarity metrics can be used as features. External plagiarism detection approaches mostly follow a binary classification model. Similar to the filtering stage, if the pair is found to match certain criteria and it is over a set threshold, it will be classified as plagiarised. In other words, the similarity score will determine which class the pair will be classified as. Hence, “detection” refers to producing a decision (plagiarism/non-plagiarism) for a given case pair.

The classification of pairs in the corpus is then evaluated against the gold-standard labels given for that corpus. This usually entails the calculation of metrics such as precision, recall, F-score and accuracy, which leads to a quantitative analysis of the approach. The existing evaluation approaches are discussed in Section 3.3 and the approach applied in this thesis is described in Section 4.5.

2.5 Plagiarism Direction Detection

The detection of plagiarism direction is a fairly new research field. Related research is discussed in Section 3.4. Plagiarism direction refers to the task of distinguishing between original and plagiarised texts without comparing them directly. The hypothesis is that source and plagiarised texts exhibit significant and measurable differences, and that these can be captured through statistical and linguistic indicators.

This is a different task to external plagiarism detection, as its requirements are very different from the traditional three-stage approach, and focuses on the traits that fit into specific patterns. Each case is usually a segment of text, comprising several sentences to paragraphs. The task of distinguishing original from plagiarised texts can be tested by binary classification or pair-wise ranking, and the text segments will be treated as individual cases rather than a document pair, which means each case will have its own classification. This approach brings a novel perspective to the plagiarism detection field, as it does not rely on the pair-wise comparison between many suspicious cases and many source cases. If the data collection is very large, traditional approaches will require more computational resources, whereas this approach relies on fitting each case into a pre-defined model. This can be applicable in a number of other research areas, for example intrinsic plagiarism detection and authorship identification, where a pattern is built for each author profile and the task is to fit each case into a specific pattern.

The general framework of plagiarism direction detection can be described as a two-stage approach: feature extraction and classification. The first stage prepares the dataset and extracts features that best represent the traits of plagiarism. The selected features are included as training and testing data in the second stage, which is to apply classification and ranking algorithms to the feature sets. The classification of each case will then be evaluated against the baseline, using standard metrics such as precision, recall, F-score, and accuracy for quantitative analysis. This general approach provides the foundation of the proposed framework described in Chapter 7.

Chapter 3

Previous Work on Plagiarism Detection

This chapter describes existing plagiarism detection methodologies. In Section 3.1 the main focus is on external plagiarism, but it also briefly covers some related research on other plagiarism types. Section 3.2 lists current research on the role of Natural Language Processing in plagiarism detection. Section 3.3 describes existing evaluation approaches, and Section 3.4 explores related fields and how they may help with plagiarism detection. Finally, Section 3.5 explores the limitations of existing plagiarism detection methodologies and the challenges faced by this research.

3.1 External Plagiarism Detection

We reiterate that the goal of plagiarism detection approaches is to identify potentially plagiarised-source pairs. A system determines which case pairs are likely to be plagiarised by analysing the similarity levels between texts in the dataset. If the similarity level between a case pair is high, the system indicates the case pair is suspicious and suggests to the user that this pair may require further investigation.

3.1.1 Early research

Plagiarism detection systems started off as detection tools for multiple-choice tests (Angoff, 1974) and computer source code (Ottenstein, 1976). Plagiarism detection systems for natural language were not developed until the 1990s.

Between 1990 and 2000, most systems developed were intended for detecting programming code plagiarism, and only a handful of researches focused on plagiarism detection for written texts. An example of these early written text detection approaches was a prototype, COPS. It was designed to detect complete or partial copies of digital documents (Brin et al., 1995). The similarity between documents was measured by using sentence-level matching. The sequences of sentences in each document were matched against other sequences in documents in the dataset. However, the sentence-based approach was not effective in detecting partial sentence overlaps.

As an extension to COPS, Shivakumar and Garcia-Molina (1995, 1996) proposed a prototype SCAM. The approach introduced the removal of stopwords and frequent words as a pre-processing step. The texts were compared as overlapping sequences of words or sentences, and thresholds were set to determine three levels of overlap between texts: exact copies, high overlap and some overlap. The results have shown that using sequences of words as a feature led to better accuracy, and the removal of stopwords has been suggested as a direction for further study. Setting a similarity threshold is a common filtering and detection approach, which is also adopted in this study.

Another early example of research, the YAP3 tool (Wise, 1996) for identifying similarities in programming code, utilised the Running-Karp-Rabin Greedy-String-Tiling (RKR-GST) algorithm as a structured-metric similarity detection system. The RKR-GST algorithm is a variant of the Longest Common Subsequence (LCS) algorithm which allows a maximal match alongside a minimal match length between texts. This algorithm was introduced to handle cases where sequences of texts had been reordered. The tool was mainly tested on computer source code and further experiments were suggested to evaluate the effectiveness of the RKR-GST algorithm on written texts.

By the end of the decade in 2000, only a handful of commercial approaches were available (Lathrop and Foss, 2000) for written text plagiarism detection, for example EVE2 and iParadigms (the early version of Turnitin©). Both approaches perform online detection by searching for similar texts on the Internet, and offline detection by comparing texts with their own database.

3.1.2 Recent research

Between 2000 and 2012, the field saw a huge surge of new plagiarism detection methodologies and their implementations. From 2000 onwards, more and more research began to address the issue of written text plagiarism detection. The exact algorithms of many commercial tools are not known, whereas the general approaches for existing plagiarism detection research are mainly non-NLP based. Although more advanced plagiarism detection methods emerged during this period, detection approaches are still not sufficient to deliver a final verdict and human judgement is always required in the end (Lukashenko et al., 2007).

Clough (2000) gave an overview of plagiarism tools and technologies. The report highlighted several related fields that may contribute to plagiarism detection. For example, the project Measuring Text Reuse (METER) (Clough et al., 2002a) investigated the reuse of texts in journalism, which may also be applicable to plagiarism detection as both applications explore paraphrases in texts. Other fields such as forensic linguistics, computational stylometry and authorship attribution, that explore approaches of text similarity detection, may also be beneficial to plagiarism detection. More details on related fields can be found in Section 3.4 of this thesis. Clough (2003) also explored the nature of plagiarism and gave an overview of issues of multilingual plagiarism detection. The report suggests using Natural Language Processing techniques and machine learning methods as future improvements to the plagiarism detection task.

A technical review of early plagiarism detection systems by Bull et al. (2001) described five systems and made recommendations to the Joint Information Systems Committee (JISC). The five systems are: Turnitin©, Findsame, EVE2, CopyCatch and WordCHECK. The report recommended further trials of EVE2, CopyCatch and Turnitin© for their ability to handle a large amount of documents, and also to evaluate their effectiveness in detecting from multiple sources.

Furthermore, Chester (2001) provided a pilot study of Turnitin© in an educational setting for the JISC. As a consequence, JISC recommended the online commercial detection tool Turnitin© as the educational tool for plagiarism prevention and identification for all higher education institutions in the UK. However, general user feedback on the tool is not satisfactory, as Turnitin© is not able to handle paraphrased texts effectively (Marsh, 2004; Weber-Wulff, 2008; Williams, 2002).

To give plagiarism detection systems a clearer classification, Lancaster and Culwin (2003) classified the systems by the type of detection methodology, availability of the system, number of documents that the metrics can process, and complexity of metrics. The types of plagiarism detection systems described in this thesis are classified on a similar basis. They concluded that rather than using the more accurate multi-dimensional metrics with large structural complexity, pair-wise metrics with low complexity are most widely adopted. This is due to the trade-off between processing resources and accuracy. The more complex the metrics are, the more processing power is required. It often takes tremendous time and effort even with the aid of powerful computers to perform detection tasks. This is not ideal for users equipped with personal computers.

The survey of plagiarism by Maurer et al. (2006) gave a comprehensive account of the plagiarism challenge. Besides reviewing some plagiarism detection systems, including Turnitin© and Copycatch, etc., they also evaluated how paraphrasing renders these tools less useful. Maurer and Zaka (2007) showed in their research that existing commercial tools were not able to cope with synonyms, resulting in many plagiarism cases that will go undetected due to paraphrasing. They highlighted issues such as extensive paraphrasing and cross-lingual plagiarism. They further suggested the use of an efficient algorithm to filter a large document collection, and then a fine-grained algorithm to be run on the reduced document collection. This allows the feasible application of deep and computing-intensive techniques. This suggestion formed the base of a five-stage filtering approach in the experimental set-up described in Chapter 4.

Another survey by Kohler and Weber-Wulff (2010) showed an astonishing growth of commercial plagiarism detection systems available online, from just five systems in 2000 to 47 in 2010. From 2004 onwards, Weber-Wulff[8] has been testing plagiarism detection systems using a small collection of manually created test cases. By using manually created plagiarism cases (short essays between one page and 1.5 pages) The effect of using umlauts in other languages (such as the German alphabet *ä*) was tested along with the extent of direct copy, translated plagiarism and collusion. The tests are based on a set of test cases, including genuine student plagiarism cases and manually created cases. This testing method is repeated over a period of six years, and in the most recent test (2010) 26 systems were tested with 42 German, English and Japanese cases. Each system is graded by the level of effectiveness, usability and professionalism. An evaluation on the system's usability showed that most commercial systems only reached the level of "barely useful", while the more well-known systems such as Turnitin© are "partially useful". The major concern is that plagiarism systems can only detect verbatim copies, and not paraphrases. It is also worth noting that the use of 3-grams and 5-grams are the standard practice in these systems.

An approach to detecting text reuse is described in Piao et al. (2001). It highlighted three important characteristics for text reuse, which are: inflectional

[8] http://plagiat.htw-berlin.de/software-en/

change, synonym substitutions and word-order change. It covered text pre-processing including stemming and the use of a sample thesaurus of synonyms. The matching sequence is visualised using Dotplot. It identifies the matching sequence on charts by using dots to display density of overlaps.

A plagiarism detection approach is described in Fullam and Park (2002). The importance of text pre-processing is stressed, where common words in suspicious and source texts are removed and every word is stemmed to its root. Similarity comparison is performed at the sentence level using the Cosine metric, and if the similarity score of a sentence pair is higher than a given threshold, the pair is marked as plagiarised.

To tackle the external plagiarism detection task, Culwin and Lancaster (2001) developed a prototype of a detection system that is capable of visualising the segments of copied texts between two documents. However, the system is not able to handle plagiarism from multiple sources. Further to the initial research, Lancaster and Culwin (2004b,a) performed tests on several matching methods and argued that n-gram matching is the best method. They also described their plagiarism detection tool, PRAISE, which uses n-gram matching.

One of the most effective detection approaches is the n-gram overlap method, which is based on calculating the amount of common word sequences between texts. N-gram overlap has been applied in other fields, such as text categorisation using 2-grams of words (Tesar et al., 2006). Similarities between texts are determined by distance or similarity metrics such as Euclidean and Cosine distance, Jaccard index, and Dice coefficient. These metrics give similarity scores and rank documents according to their level of resemblance. For example, Monostori et al. (2000) developed the MatchDetectReveal system to identify direct copies of written texts, using a simple string-matching algorithm. N-gram overlap methods are able to identify direct copies, but they are ineffective if the plagiarised texts involve more complex changes such as paraphrasing. In Monostori et al. (2002) they extended their research by suggesting segmentation strategies of various lengths in the comparison stage. Detection is based on overlapping n-grams of characters between documents, which would not be ideal in complex plagiarism situations.

Another example of the n-gram overlap method is the use of overlapping 3-grams in the Ferret plagiarism detector (Lane et al., 2006; Lyon et al., 2001, 2006). The methodology pre-processes documents into sets of 3-grams of words, and then compares each set between document pairs. The similarity score is determined by the Jaccard coefficient (Formula 4.1 on page 103) and a variation of the Jaccard coefficient where the number of matching 3-grams is divided by the number of distinct 3-grams. It is said that 3-gram is the optimal n-gram size for matching shorter documents which consist of minimal paraphrasing.

White and Joy (2004) suggested comparing documents at the sentence level, and also described the use of text pre-processing techniques such as tokenisation, lowercasing, punctuation and stopword removal. Their sentence-based algorithm can detect direct copies, as well as paraphrasing and sentence-level changes. However, the algorithm only calculates the number of words in common and the average length between the sentences. Any changes crossing the sentence boundaries are difficult to detect.

The use of n-gram matching is again explored in Nahnsen et al. (2005). Their method involved the comparison of sequences of lexically-generalised words (lexical chains), and similarity is computed by using cosine similarity on the lexical chains and the term frequency-inverse document frequency (tf-idf) of weighted keywords. The tf-idf reflects how important a word is to a document in the dataset. To reduce the number of false cases, they further introduced the use of syntactic information in plagiarism detection, which is described in Section 3.2.

Bao et al. (2004) also used n-gram matching in their work, which computes similarity by counting the frequency of common words between two semantic sequences. A semantic sequence refers to a sequence of words with stopwords and non-frequent words removed. Their method is very effective for detecting direct copies but it does not perform well at detecting re-worded plagiarism.

To deal with the challenge of large-scale document collection, an improvement to the filtering stage was proposed in Barrón-Cedeño and Rosso (2009). They proposed to use Kullback-Leibler symmetric distance as a filtering strategy to reduce the search span. Kullback-Leibler distance measures the closeness of two probability distributions. The probability distributions contains terms from the case pairs, which are selected by features such as tf-idf. The experiment showed that 2-grams are better for enhancing Recall whereas 3-grams are better for enhancing Precision, and the best results were obtained with 1-grams. Their work is focused on reducing the search space, but in order to measure its effectiveness, further work is needed using a larger corpus, as the experiment was performed on a relatively small text reuse corpus with 700 documents.

To overcome the lack of realistic plagiarism test cases, Clough and Stevenson (2010) developed a corpus of plagiarised short answers using a manual method. They also used similarity metrics to calculate the amount of overlaps of matching word n-grams. The evaluation metric is described in Section 3.3, and their corpus, which is used in this thesis, is described in Chapter 5.

Other structural methods such as research on Multilevel Text Comparison (Zini et al., 2006) and Plagiarism Pattern Checker (Kang et al., 2006) are also used in plagiarism detection. The research by Zini et al. (2006) explored the use of n-gram matching, but they also proposed to measure the edit distance for each 4-gram of segments. Their method of multilevel text comparison looks into various levels of the document structure, and calculates the matching proportion of exact sentences and word sequences in document pairs. Chunks of a lower level refers to words, and chunks of higher level refers to paragraphs. The similarity between document chunks is calculated using Levenshtein Edit Distance to determine the amount of insertion, deletion or substitution between texts. The weight of each chunk correlates with the level of structure , and a threshold is set to filter chunks that have been given lesser weight. However, no substantial experimental results have been reported that evaluate the effectiveness of the algorithm.

Kang et al. (2006) developed another structural metric, Plagiarism Pattern Checker, which checks for plagiarism patterns by measuring the amount of overlapping n-grams within a sentence. It also incorporates WordNet to check for synonyms. They claim that the incorporation of lexical generalisation is a contributing factor to a more precise plagiarism detection approach, and this will be further explored in this study.

Fingerprinting methods started to gain attention as the amount of information available increased. This method is said to be much more efficient as it generates a hashed description, which is the "fingerprint" for each document, and then the fingerprints of documents can be compared instead of the entire document. Hoad and Zobel (2003) developed a document fingerprinting method for plagiarism detection by using word frequency. For example, phrases of 3-grams or 4-grams are selected, and then their frequency distribution forms the fingerprint. This may allow a quicker n-gram comparison but the question of complex structural changes is left unanswered. Similarly, n-gram matching and fingerprinting is used in the KOPI online plagiarism detection tool, detailed in (Pataki, 2006). The paper described a six-step approach that condenses to pre-processing, fingerprinting, and matching.

The importance of reducing false positives without affecting true positives is emphasised in research by Sorokina et al. (2006). They implemented a large-scale plagiarism detection method for a research document collection. They used 7-grams of words and fingerprinting in their system. The fingerprint is a representation of each document obtained by summarising it with a small set of character sequences. The comparison is then based on finding matches between the sets of fingerprints instead of actual texts from the document. This research also drew attention to the lack of a good-quality plagiarism detection corpus: as there is a lack of concrete cases, they had to use "likely plagiarism cases" where it is difficult to obtain an accurate annotation.

The combination of fingerprinting and Vector Space Model (VSM) was described in Stein et al. (2007b) as part of a three-stage approach to retrieve plagiarised documents. VSM represents words in a multi-dimensional vector where each word corresponds to a dimension of the space. The frequency of words is weighted using different strategies, for example term frequency and/or inverse document frequency (tf-idf). The position of the word in the vector correlates to the weight given. This gives the VSM an advantage over the use of a similarity metric, as VSM measures the level of term occurrences as well as the similarity between texts. However, it does not handle paraphrased texts and the word order would be lost using this representation.

While most research targeted the plagiarism detection challenge by developing a complete framework, other research aimed to tackle certain stages of the traditional detection pipeline. Stein et al. (2007b) introduced a three-stage approach tested on chunks of the Wikipedia corpus, using n-gram hash-based indexes to create fuzzy fingerprinting for retrieving documents, which aims to speed up candidate document retrieval from a large corpus. The experiment shows that indexing can improve the efficiency of the candidate document retrieval stage, but the analysis was not based on a plagiarism corpus and it lacks a thorough discussion on how realistic plagiarism cases will affect such indexing techniques.

One of the earliest plagiarism detection systems using stylometry features is proposed by Gruner and Naven (2005). The method involved text pre-processing, splitting documents into chunks of text, and then analysing the word pattern ratio of each block. The word pattern ratio is an adaptation of the non-contextual measurements proposed in authorship attribution studies, such as “fraction of all sentences with ‘a’ in which ‘a’ is the first word of the sentence”. The score of the word pattern is then matched against another block to calculate the level of similarity. If the number of matches reached a pre-determined similarity threshold, the block of text is considered as plagiarised.

Other advanced approaches started to emerge by 2007. The research by Rehurek (2007) suggested using a semantic-based approach for plagiarism detection, by combining an information retrieval model based on tf-idf with latent semantic indexing (LSI). The bag-of-words approach at the document level can represent the documents better as the feature is not limited by the sentence boundaries. The LSI technique is based on VSM that aims to analyse the conceptual relatedness between documents, exploring the structure of words that co-occur together. No experimental result is given that evaluates the proposed approach.

Dreher (2007) suggested the use of a Normalised Word Vector algorithm to measure similarity, which is based on the VSM with synonym generalisation performed on each word. However, even with the emergence of more advanced approaches, paraphrases remained a challenge.

In Pera and Ng (2009), text pre-processing techniques, such as stopword removal, and shallow NLP techniques, such as stemming, are applied to documents before computing similarity. Short sentences are also removed. The degrees of similarity between words are calculated by their frequency of co-occurrence and relative distance, as denoted by a word-correlation matrix generated using Wikipedia. A threshold is set to filter sentences with a low similarity, and the degree of resemblance between two documents is visualised using Dotplot view. Although the results showed improvement over n-gram matching by reducing the false positives, the approach is still limited to comparison between individual words.

In recent years, combining similarity metrics with information retrieval models has become a common approach in the field. For example, the use of similarity metrics with VSM was investigated in Tsatsaronis et al. (2010). Their research stated that statistical metrics are used in plagiarism detection methods as they allow simpler implementation and are effective against verbatim plagiarism, but they will not aid the semantic analysis of textual and non-textual information.

The characteristics of paraphrasing are explored in Sousa-Silva et al. (2010), who performed a small-scale analysis on five Portuguese documents using a forensic linguistic approach to plagiarism detection. The research showed that replacing words with semantically-related words, e.g. synonym substitution, is a major feature that suggests a case of paraphrasing. Other features that can confuse a plagiarism detection system include insertion of words and change of word order.

3.1.3 PAN workshop and competition series

To address the increased attention in the field, the first workshop of "Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection" was held in conjunction with the 30th Annual International ACM SIGIR conference (Stein et al., 2007a). The workshop focused on three tasks: 1) Plagiarism analysis, 2) Authorship identification and 3) Near-duplicate detection. The workshop acted as the pilot of the first PAN plagiarism detection competition in 2009, which will be described shortly. In 2007, only one submission discussed the challenge of intrinsic plagiarism detection (explained in Section 3.4). Five submissions focused on authorship identification, which are also described in Section 3.4. One submission discussed the identification of near-duplicate music documents which used melody and music theory as features (Robine et al., 2007). The workshop concluded that it is necessary to segment long texts in a document to chunks, and raised two main issues: 1) the lack of a benchmark corpus to evaluate plagiarism detection systems, and 2) the lack of an effective plagiarism detection tool that does not trade off computational cost with performance.

Following the success of the 2007 workshop, in 2008 another specialised workshop on "Uncovering Plagiarism, Authorship and Social Software Misuse" was held in conjunction with the 18th European Conference on Artificial Intelligence (Stein et al., 2008b). The focus of the workshop was defined in three tasks: 1) Plagiarism analysis, 2) Authorship identification and 3) Social software misuse. Near-duplicate detection was replaced by social software misuse, which refers to the problem of anti-social behaviour in online communities.

One study described the use of statistical alignment models in the task of cross-lingual plagiarism detection (Barrón-Cedeño and Rosso, 2008), described in Section 3.4. The paper also described a preliminary experiment on external plagiarism detection using statistical language models on three aspects: word, POS and stem. Statistical language models trained on original words, part-of-speech of words and stemmed words provided a platform to analyse sequences of tokens. The result suggested that further experiments should combine the three aspects instead of analysing them separately.

One paper described an indexing approach for information retrieval used for plagiarism detection. It pointed out the need to find the trade-off between precision and recall to suit various tasks (Creswick et al., 2008). Another paper presented two approaches to distinguish natural texts from artificially generated ones, which can be applied in tasks such as detecting spam emails. The first approach used language models and the second focused on using relative entropy scoring, which gives higher weight to n-grams which exist in the Google's n-grams model (Lavergne et al., 2008).

The third PAN workshop on "Uncovering Plagiarism, Authorship and Social Software Misuse" was held in conjunction with the 25th Annual Conference of the Spanish Society for Natural Language Processing (Stein et al., 2009). The aims of the workshop remained the same as the 2008 workshop. Different from previous years, the workshop was co-organised with the first International Competition on Plagiarism Detection. The focus was shifted from bringing together theoretical research in the field to a more competitive development workshop. The competition consisted of two subtasks: external plagiarism detection and intrinsic plagiarism detection. There were a total of 13 groups participating in the competition. The competition was based on a large-scale artificially created plagiarism corpus and provided an evaluation framework for plagiarism detection. Nine groups entered in the external plagiarism detection task and three groups entered in the intrinsic plagiarism detection task, with one group entering in both tasks. The second and third competitions are more mature and hence will be explained thoroughly in the following subsection. The details of the evaluation framework are described in Section 3.3, and the PAN-PC-10 corpus which is used in this thesis, is described in Chapter 6.

There has been a further increase in plagiarism detection research between 2010 and 2013. With the increased interest in plagiarism detection, plagiarism detection competitions have been continually organised to encourage development and evaluation of detection systems. Following the first International Plagiarism Detection competition, a series of PAN plagiarism detection competitions were organised, with the second and third PAN competitions on plagiarism detection (Stein et al., 2009; Potthast et al., 2010c) attracting 18 and 11 participating groups respectively. The corpora used in the competitions were created with automatic insertion of texts from source texts to suspicious texts. Some of the cases involve translated plagiarism and some cases contain various levels of obfuscation, which are either artificial or manual text operations aiming to imitate paraphrasing. The evaluation is based on the standard metrics of precision, recall and F-score, and two specific metrics: granularity and overall score (these metrics are explained in Section 3.3). In a nutshell, granularity measures the accuracy of the system in finding the exact plagiarised segments, and the overall score is combination of F-score and granularity. No baseline was set for the external detection task. A detailed description of the PAN corpora and associated evaluation metric is given in Section 3.3 and Chapter 6.

Most of the participants in the competitions focused on external plagiarism. In the second competition (PAN-PC-10), there were only three systems which explored intrinsic plagiarism detection, with one system developed solely for intrinsic detection, and two systems developed for both external and intrinsic detection, compared to 17 external plagiarism detection systems. Although some levels of intrinsic detection using techniques from authorship identification and stylometry emerged in the competition, their accuracies are yet to reach a satisfactory level, as only one system performed better than the baseline, where the baseline assumed everything belongs to the plagiarised class.

In Nawab et al. (2010)'s attempt in the PAN-PC-10 competition, n-gram matching is used as the filtering metric and the Running-Karp-Rabin Greedy String Tiling algorithm is used in detailed analysis. The use of n-gram filtering is similar to the proposed framework in this study. One of the biggest challenges is the difficulty of accommodating a parameter that is specific enough to identify various levels of obfuscation in the detailed analysis stage, but general enough so that source documents are not overlooked in the filtering stage.

The same situation was observed in the third competition (PAN-PC-11), where seven systems participated in external detection, two systems participated in intrinsic detection, and two systems participated in both tasks. According to the organisers, the PAN-PC-11 corpus features plagiarism cases which are more difficult to detect, as it is clear that verbatim plagiarism does not pose enough of a challenge. Therefore, the PAN-PC-11 corpus features more manually or artificially obfuscated cases. The results from the competition show that there is a drop in performance, which indicates that obfuscation does pose a better challenge to plagiarism detection systems and that there are no good enough techniques that can tackle paraphrasing.

In the 2012 PAN workshop(Potthast et al., 2012), 15 teams participated in the external plagiarism detection task. Two sub-tasks are introduced, which include candidate document retrieval and detailed document comparsion. Seven teams re-used their systems from previous PAN competitions. New approaches to detect similarities in the detailed comparison stage include sequence alignment algorithms which are applied in the bioinformatics field. Other developments suggest that a one-fits-all approach is not ideal, but an adjustable approach poses a challenge to current research.

In general, the external plagiarism detection task participants in the workshop series can be summarised as taking a three-stage approach: *pre-processing*, *detailed analysis* and *classification*. The **first stage**, *pre-processing*, is done by processing the document collection using stopword removal, synonym replacement and stemming, then transforming the document into hashed word n-grams. The source documents are processed as an inverted index and compared with the suspicious documents by using a metric similar to the Jaccard coefficient. This filtering stage is essentially narrowing down the search span of suspicious-source document pairs. The **second stage**, *detailed analysis*, investigates the candidate suspicious-source document pairs. This is usually done by using heuristic sequence alignment algorithms or similarity scores from n-gram overlap counts. The **third stage**, *classification*, aims to reduce the number of false positive detections. This is done by applying heuristics such as setting a minimum length of passage detected, or a threshold on the similarity score.

To conclude the approaches used in the PAN competition, it is important to note that most approaches employ brute-force pair-wise matching, and that the use of word 5-grams contributed to the winning approach in 2010 . The participants do not apply any deep natural language processing techniques to search for the deeper linguistic information, which is needed for handling paraphrases. Although some approaches employed shallow language processing techniques, the benefit of NLP in plagiarism detection is left unexplored. Therefore, one of the goals of this thesis is to explore the benefits of individual NLP techniques as well as the most favourable combination of NLP techniques.

Another note is that although precision of the PAN systems is very high, recall is generally low, with the exception of recall on verbatim copies which is higher. The competition indicated that manual obfuscation which includes paraphrases poses a far greater challenge than artificially obfuscated texts. Hence, another goal of this thesis is to improve recall on manually paraphrased texts.

3.1.4 Summary

To summarise this section on existing plagiarism detection approaches, the general non-NLP based approaches for existing plagiarism detection research are grouped as follows:

1. Overlapping word n-grams
2. Frequency-based method
3. Fingerprinting
4. Structural method

The first group is a superficial approach that is based on pair-wise comparison between texts. Texts in the dataset are extracted as sequences of n-grams, and the similarity between texts is calculated by applying a similarity metric. For example, the Jaccard coefficient is a metric which normalises the amount of overlapping n-grams with the union of n-grams in both texts (Nahnsen et al., 2005; Zini et al., 2006; Lancaster and Culwin, 2004a).

The second group is based on a statistical approach which calculates the weight of word distributions across documents. The weight is determined by features such as tf-idf, and variations which consider the document length and frequency of term in the document are often used. This method is based on the hypothesis that similar documents should contain words with similar number of occurrences (Hoad and Zobel, 2003).

The third group aims to produce a description (a "fingerprint") for each document in the collection. The fingerprint presents the document and comparison is based on the fingerprint instead of the acutal document, thereby reducing the need to perform exhaustive comparison (Shivakumar and Garcia-Molina, 1995). Substrings from the document are converted as hashed index for subsequent querying. The strategies of selecting the substrings vary, which include full fingerprinting, positional selection, frequency-based selection and structural-based selection.

The fourth group is based on structural methodes, which identifies patterns between the query and collection based on indexing and retrieval metrics such as Latent Semantic Indexing (LSI). LSI is able to analyse the similarities between texts based on their contextual meaning, where the underlying structure in the word usage corresponding to the document is represented by associating words in similar contexts (Ceska, 2009).

All of the research listed in the above sections shares a common plagiarism detection process which involves the following three-stage approach:

1. Pre-processing: Apply text processing and basic text processing techniques to the text collection, and then perform candidate retrieval by data filtering of suspicious-source pairs using a simple similarity metric.
2. Detailed analysis: Apply deeper techniques on the candidate pairs extracted from Stage 1, repeat the filtering process and narrow down the search span of candidate pairs.
3. Classification: Use similarity scores from Stage 2 to give each candidate pair a classification, either by setting a threshold or by using a machine learning algorithm.

As this project requires the application of various techniques, additional stages to perform filtering and further processing are needed. The three-stage approach forms the basis of the proposed five-stage approach described in Chapter 4.

Overall, to date, the most widely adopted plagiarism detection approaches, including those listed above, are still based on pair-wise comparison that only investigates superficial features of texts. This is due to the trade off between processing resources and accuracy. The more complex the approach is, the more processing power is required and it would often take tremendous time and effort with the aid of super computers to perform detection tasks. This is not ideal for users equipped with personal computers, thus online systems like Turnitin© offer a platform for users to upload their documents onto the server. The comparison is not performed locally but on a remote server. However, system complexity remains an issue as more elaborate metrics require extensive processing time and resources.

Nevertheless, recent research – including that proposed in this thesis – has considered the use of NLP techniques to aid the investigation of more elaborate similarity metrics in plagiarism detection. The aim is to utilise techniques that can extract the underlying syntactic and semantic information of texts to analyse complex plagiarism cases, especially those cases where overlapping word n-grams and word frequency-based methods are incapable of detection.

3.2 Natural Language Processing in Plagiarism Detection

This section summarises the existing work on using NLP for plagiarism detection and introduces the techniques which will be investigated in this research. NLP involves the processing of human languages by computers. Many fields such as computer-assisted language learning (Chang and Chang, 2004), extraction of biomedical information (Terol et al., 2004) and search engine optimisation (Penev and Wong, 2006) have already experienced benefits from using NLP. However, it remains an under-explored area for plagiarism detection.

Previous work by Clough (2003) suggested applying NLP techniques for plagiarism and that this could yield better accuracies through the detection of paraphrased texts. Although no experiments were performed to show that this was indeed the case, this work has inspired the use of NLP in the plagiarism detection field.

In all plagiarism detection systems, pre-processing and candidate filtering are essential tasks. Pre-processing allows the generalisation of texts, and candidate filtering reduces the search span for further analysis stages to optimise performance. This is particularly important when a large number of documents are involved. The NLP techniques described in this section are applied in various stages. Most commonly, shallow NLP is applied in the text pre-processing stage, whereas deep NLP is applied in a deeper analysis stage.

Shallow NLP techniques refer to simpler, low resource-demanding techniques, such as tokenisation, lowercasing, stopword removal, lemmatisation and stemming as part of the pre-processing stage. For example, Chuda and Navrat (2010) proposed the application of tokenisation, stopword removal and stemming to Slovak texts in the text pre-processing stage, but the individual contribution of each technique was not fully investigated and the system was not tested extensively.

Similarly, Ceska and Fox (2009); Ceska (2007, 2009) proposed the incorporation of latent semantic analysis along with text pre-processing techniques for plagiarism detection. The actual comparison is still limited to n-gram matching by singular value decomposition, which involves the retrieval of truncated singular values and vectors from an original term-document matrix. These techniques involved simple heuristics including replacement of numbers with a dummy symbol, removal of punctuation, application of basic NLP techniques such as lemmatisation, removal of irrelevant words and incorporation of a thesaurus to generalise the words in the texts. While some of the heuristics had a positive impact on the accuracy of their plagiarism detection approach, the use of NLP techniques did not show significant improvement with respect to the word n-gram overlap approach. It is believed that this is due to the limitations of both the NLP techniques used and their experimental settings, including the use of small corpora and inaccurate disambiguation procedures for generalising words. To address this challenge, the proposed framework (Chapter 4) in this thesis combines text pre-processing and NLP techniques with a machine learning classifier.

The application of shallow NLP techniques such as tokenisation, lowercasing, punctuation removal, stopword removal, lemmatisation, stemming and part-of-speech tagging in the proposed framework is described in Section 4.2.

Using deeper NLP techniques to investigate the structure of texts rather than their superficial information, Leung and Chan (2007); Mozgovoy et al. (2007) suggested using parse trees to find the structural relations between documents.

The research by Uzuner et al. (2005) used shallow semantic and syntactic rules to capture traits of rewriting. The semantic class of each verb is determined by a part-of-speech (POS) tagger and the syntactic structures are extracted for each sentence. A semanitc class represents a group of verbs which are similar in meaning. The similarity matching is not based on words, but on the verb classes, thereby matching synonyms that retained the same word order. The experiment is performed on translated texts from 49 books, which represent different levels of paraphrasing. The results showed that syntactic features can achieve better performance than tf-idf, and that linguistic techniques can help to identify paraphrases better than statistical methods. Although the results are promising, the nature of the corpus used in the experiment is different from plagiarism, in the sense that translated books will follow the same sentence structure as the original, whereas sentence-level paraphrasing and more complex text operations will often be seen in plagiarised texts.

Mozgovoy et al. (2006) described an approach to apply text pre-processing and NLP techniques to a plagiarism detection system for the Russian language. The techniques include tokenisation, generalisation of words into their hierarchical classes such as substituting the word “fox” with “animal” , and extraction of functional words and argumentative words for matching. Mozgovoy (2007) also provided an insight into the study of plagiarism detection. The suggestion is to improve string matching algorithms by incorporating tokenisation and syntactic parsing into written text plagiarism. However, the trade-off between efficiency and effectiveness resulted in the development of a fast string-matching algorithm rather than a deep and complex linguistic-based system.

The application of a two-stage approach to plagiarism detection is further explored in Mozgovoy et al. (2007), who proposed the use of natural language parsers to analyse the syntactic structure of texts. The first stage was to parse all documents in the dataset, and the grammatical relations generated by the Stanford Parser[9] were post-processed into groups of words. The second stage was to compute the amount of similar grammatical relations between documents. Initial experiments suggest parsing may be practical for detecting sentence re-ordering, but it is not capable of detecting paraphrases. It is proved to be feasible to use a parser in pre-progressing stages; however, their approach to parsing results in a loss of the original word order in every sentence and it is difficult for their detection system to highlight similar blocks of text. Moreover, the corpus used in their experiment is based on journalism text reuse rather than plagiarism.

A theoretical study by Leung and Chan (2007) suggested incorporating both shallow and deep NLP in automatic plagiarism detection, involving the application of synonym generalisation and extraction of syntactic structure. Semantic processing identifies the deep structure of a sentence by converting parse trees into case grammar structure. This approach compares sentences at semantic level. How-

[9] http://nlp.stanford.edu/software/lex-parser.shtml

ever, no experiments have been carried out to evaluate the actual performance of the techniques, due to the lack of a semantic analysis tool and a suitable corpus.

The solution to the problem of paraphrasing and the concept of using a thesaurus to generalise synonyms are described in studies by Ceska (2009) and Alzahrani and Salim (2010). Their experiments were performed using a Czech thesaurus and an English thesaurus respectively. For English, the authors used WordNet[10] a well-developed thesaurus which is semantically structured. It provides information on relationships between words, which allows the matching of synonyms and hyponyms. For most content words in texts, WordNet has one or more synsets (a group of synonyms) which have the same meaning as the original word. The matching of WordNet synsets with the correct sense becomes the main challenge. Chen et al. (2010) applied synonym, hypernym and hyponym substitutions using WordNet and incorporated these into ROUGE (Lin, 2004), a metric which measures similarity by n-gram frequency, skip-bigram and longest common subsequence. Although similarity metrics like ROUGE can handle simple text modification, and WordNet can handle some level of word substitution, the challenge of a higher level of paraphrasing is yet to be addressed, and the issue of Word Sense Disambiguation (WSD) with WordNet brings a major challenge. In our proposed framework, WSD is bypassed as all synsets are used, without the need to determining a specific sense for each word.

To sum up, the use of NLP techniques in plagiarism detection is still underexplored. To date, very limited research has been done to incorporate linguistic

[10] http://wordnet.princeton.edu/

techniques that can exploit lexical, syntactic and semantic features of texts into plagiarism detection approaches. Although shallow techniques have been included as part of the pre-processing stage, investigations involving deep techniques are still limited. Hence, the aim of this thesis is to explore linguistic features that may contribute to the plagiarism detection field and the combination of several techniques instead of relying on individual techniques.

In Chong et al. (2010) the combination of shallow and deep NLP techniques was employed in an experiment using a small-scale corpus of short plagiarised texts. Techniques generating features which do not rely on exact word matching, such as chunking and parsing, are compared against an overlapping 3-gram word baseline. In addition, language models are applied to generate probabilities for word n-grams, perplexities and out-of-vocabulary rates. A similarity metric, the Jaccard coefficient (Formula 4.1 on page 103), is applied to the extracted features to generate similarity scores for use in the machine learning algorithm. The results showed that the best performing features included a combination of word 3-grams, lemmatisation, language model perplexities and parsing. A detailed explanation is given in Chapter 5.

In addition to the initial small-scale study, Chong and Specia (2011) explored lexical generalisation for word-level matching in plagiarism detection. Lexical generalisation in this case substitutes each content word with the set of all its synsets. This is aimed at tackle paraphrasing in plagiarism cases. In contrast to other related research, the technique is applied without any WSD. Similarity comparison is carried out at the word level, which disregards word ordering, and the results were compared against an overlapping 5-gram word metric. The experiment was tested on a large-scale corpus and the results showed that lexical generalisation can help improve recall by reducing the false negative cases. A detailed explanation is provided in Section 6.1.

The use of additional deep NLP techniques such as named entity recognition, parsing for dependency relations, synonym generalisation using Wordnet synsets, predicates extraction, and verb generalisation using VerbNet are described in Section 4.2.

3.3 Evaluation Approaches

Even for humans, detecting plagiarism is a very difficult task when the writer has the intention of deceiving the reader. This difficulty carries over to the evaluation of plagiarism detection approaches. Evaluation approaches for plagiarism detection are often very subjective as there are no solid standards. The best means to evaluate a detection framework is to rely on a corpus of previously annotated plagiarised and non-plagiarised texts cases.

3.3.1 Evaluation corpora

A general approach of evaluating plagiarism detection systems is corpus-based evaluation. This normally involves the use of a set of plagiarised texts and non-plagiarised texts and the task is for the system to determine what class a particular case belongs to.

Before specific plagiarism corpora were developed, research in the field relied on other textual similarity corpus such as the METER corpus, which is a corpus for analysing journalistic text reuse, was developed by Gaizauskas et al. (2001); Clough et al. (2002b). The corpus is manually annotated with examples of related news articles.

In the plagiarism detection software test by Weber-Wulff (2008), 31 essays written in German were used. The essays were manually created, some were original texts without plagiarism, some contained machine translation, and some contained paraphrasing. The test was performed on 17 systems and concluded that none of the available systems achieved a satisfactory result. As the test cases are not publicly available and the size of the corpus is not sufficient for detailed linguistic analysis, these test cases are not used in plagiarism studies.

Clough and Stevenson (2010) developed a corpus of plagiarised short answers for evaluating plagiarism detection systems. The corpus is created manually, by computer science students rewriting five computer science-related short texts from Wikipedia. The students were instructed to rewrite the texts in three levels of plagiarism: near copy (verbatim), light revision (shallow paraphrasing) and heavy revision (deep structural changes and paraphrasing). Along with non-plagiarism cases, the corpus provides a near-realistic test base for experimental use. The corpus consists of 95 short answers that are between 200 and 300 words long. 60% of the cases are plagiarised. The use of this corpus in a small-scale experiment is described in Chapter 5.

The first PAN workshop in 2009 introduced an artificial corpus for evaluating plagiarism detection systems. Stein et al. (2009) created the PAN-PC-09 corpus which consists of 41,223 documents with 94,202 plagiarism cases. The corpus is created for both external and intrinsic plagiarism detection tasks, and some cases are designed for cross-lingual plagiarism detection.

The document length is between 1 to 1000 pages. Half of the documents are suspicious and half of them are source. Half of the suspicious documents do not contain plagiarism, thus they are the clean cases. Plagiarism cases are between 50 and 5000 words, and the majority of the cases are in English. To represent paraphrased text, artificial operations, referred as obfuscations, are inserted into the plagiarism cases. They include: random text operations, which shuffle, remove, insert or replace words at random; semantic word variation, which replaces a word with its synonyms, hyponyms, hypernyms or antonyms, by random selection ; and POS-preserving word shuffling, which shuffles words while preserving the POS sequences. These artificially-generated texts will not make any sense to a human, but with the lack of better genuine cases and for the purpose of the pilot study, the PAN-PC-09 corpus provides a sufficient test base for the candidate document retrieval task.

Similar to the PAN-PC-09 corpus, the PAN-PC-10 corpus (Potthast et al., 2010c) consists of 27,073 documents with 68,558 plagiarism cases. The general specification of the corpus is very similar to that of the previous year, with the exception that the PAN-PC-10 corpus has 6% *simulated plagiarism cases*. There are 4,067 plagiarised text passages with their corresponding source text passages. The length of the simulated plagiarised passages is between 21 and 1,190 words, and the source passages between 74 and 745 words. The *simulated plagiarism* *cases* are manually written using *Amazon Mechanical Turk*, thus they are the closest imitation to real plagiarism cases available for experimental purposes. The use of the PAN-PC-10 corpus simulated cases in this thesis is described in Chapter 6.

The PAN-PC-11 corpus consists of 26,939 documents and 61,064 cases. The general specification is similar to that of previous years, with the exception of an increased amount of total obfuscated cases from 60% in 2010 to 82% in 2011 and a slight increase in *simulated plagiarism cases* from 6% in 2010 to 8% in 2011.

The general criticism of the PAN corpora is that they lack realistic cases. Even with the introduction of simulated cases, participants do not employ linguistic techniques to deal with them, as they only represent a small part of the corpus and hence do not have much influence on the overall detection score. The majority of the systems which participated in the competition employed brute-force detection techniques to focus on the artificial cases. In the first PAN competition, the size of the corpus posed a challenge. The major focus in the second competition was the manual simulated cases. The third competition saw an increased level of difficulty in detection caused by complex paraphrasing in cases.

The PAN-PC-2012 corpus is created in similar fashion as previous years. Source cases are extracted from books of Project Gutenberg, automatically obfuscated and then inserted into suspicious cases. The types of plagiarism in the corpus include: word-for-word copy, low artifical obfuscation, high artificial obfuscation, manually simulated plagiarism, and translated plagiarism. Each of these categories contained 500 cases, and there are 500 clean cases in the corpus. There are also

33 cases of real plagiarism retrieved online, which are short texts containing 75 to 150 words. Real cases are a welcoming sight yet the number of cases is still not sufficient.

3.3.2 Evaluation metrics

The performance of a plagiarism detection system is commonly evaluated by standard evaluation metrics such as precision, recall, and F-score. In addition, two metrics have been proposed in the context of the PAN competitions (Potthast et al., 2010b): *granularity* and *plagdet*. Granularity measures the accuracy of the approach in finding the correct segmentation for plagiarism cases, and it is only appropriate for passage level detection. Plagdet represents the overall score which combines granularity with F-score.

More formally Potthast et al. (2010b), a plagiarism case within a document d_{plg} is defined as a 4-tuple which contains the start and end positions of the passage in a plagiarised document, and the start and end positions of the referenced passage in the associated source document. A plagiarism case is thus denoted as $s =< s_{plg}, d_{plg}, s_{src}, d_{src} >$, $s \in S$ is represented as a set s of references to the characters of d_{plg} and d_{src} that form the passages s_{plg} and s_{src}. Likewise, a plagiarism detection $r \in R$ is represented as r. Based on this 4-tuple, micro-averaged precision and recall of R under S are defined as follows:

$$prec_{mic}(S, R) = \frac{|\bigcup_{(s,r)\in(S\times R)}(\mathbf{s} \sqcap \mathbf{r})|}{|\bigcup_{r\in R}\mathbf{r}|} \tag{3.1}$$

$$rec_{mic}(S,R) = \frac{|\bigcup_{(s,r)\in(S\times R)}(\mathbf{s}\sqcap\mathbf{r})|}{|\bigcup_{s\in S}\mathbf{s}|} \tag{3.2}$$

The macro-averaged precision and recall of R under S are defined as follows:

$$prec_{mac}(S,R) = \frac{1}{|R|}\sum_{r\in R}\frac{|\bigcup_{s\in S}(\mathbf{s}\sqcap\mathbf{r})|}{|\mathbf{r}|} \tag{3.3}$$

$$rec_{mac}(S,R) = \frac{1}{|S|}\sum_{s\in S}\frac{|\bigcup_{r\in R}(\mathbf{s}\sqcap\mathbf{r})|}{|\mathbf{s}|} \tag{3.4}$$

where

$$\mathbf{s}\sqcap\mathbf{r} = \begin{cases} \mathbf{s}\cap\mathbf{r} & \text{if r detects s,} \\ \emptyset & \text{otherwise.} \end{cases} \tag{3.5}$$

A plagiarism case s is defined as $\mathbf{s} = \mathbf{s}_{plg}\cup\mathbf{s}_{src}$, where $\mathbf{s}_{plg} \subseteq \mathbf{d}_{plg}$ and $\mathbf{s}_{src} \subseteq \mathbf{d}_{src}$. Similarly, A detection r is defined as $\mathbf{r} = \mathbf{r}_{plg} \cup \mathbf{r}_{src}$.

Plagiarism detection is this context is defined as $rdetectss$ if $r_{plg} \cap s_{plg} \neq \emptyset$, $r_{src} \cap s_{src} \neq \emptyset$, and $d'_{src} = d_{src}$.

The metric granularity is applied to account for cases where overlapping or multiple source texts are detected for a single plagiarism case:

$$gran(S,R) = \frac{1}{|S_R|}\sum_{s\in S_R}|R_s| \tag{3.6}$$

where $S_R \subseteq S$ are cases flagged as positive in R and $R_S \subseteq R$ are possible original texts of s:

$$\begin{aligned} S_R &= \{s | s \in S \wedge \exists r \in R : \text{r detects s}\}, \\ R_S &= \{r | r \in R \wedge \text{r detects s}\}. \end{aligned} \tag{3.7}$$

These measures are combined as a specifically designed metric, pladget , for the plagiarism detection competition as the nature of the competition is both a retrieval task and extraction task. To compute an overall score, plagdet is defined as follows:

$$plagdet(S, R) = \frac{F_1}{log_2(1 + gran(S, R))'} \tag{3.8}$$

where $F1$ is the equally weighted harmonic mean of macro precision and recall.

In this thesis, the focus is not on participating in the competition, but on exploring the incorporation of NLP techniques. Therefore, the experiments described in this thesis do not focus on the passage segmentation problem, which makes it impossible to compute the granularity and plagdet metrics. The detailed experimental set-up is described in Chapter 6.

3.4 Related Fields

This section describes other fields that are related to plagiarism detection, with techniques that may also be beneficial to the development of plagiarism detection approaches.

3.4.1 Authorship identification and intrinsic plagiarism detection

Authorship identification is sometimes referred as authorship attribution and verification. Authorship attribution is the task of relating pieces of anonymous texts to potential authors, based on examples of other texts by the same authors. On the other hand, authorship verification is the task to determine whether or not pieces of texts belong to one given author.

Authorship identification is a task that is closely related to intrinsic plagiarism detection. Authorship attribution is often treated as a text categorisation task, for example, the authorship attribution system by Nazar and Pol (2006) which categorised texts based on 2-grams of words alone. Another example is the character n-gram-based authorship attribution by Keselj et al. (2003), which was applied to Chinese and Greek texts. These studies can be considered a language independent detection approach, as they are based on matching of n-grams and no other features. Coyotl-Morales et al. (2006) used word n-grams overlap in their authorship attribution approach. Their method characterises documents by frequency of the sequence of function and content words. Their argument is that using a simple frequency-based approach is better than using sophisticated linguistic analysis of text.

A comparison by Grieve (2002) shows that the best approach involves the analysis of many types of textual measurements, including the frequency of common words and punctuation marks, and character-level n-grams.

Zheng et al. (2006) used lexical, syntactic, structural and content-specific features to identify authorship. Lexical features included average word/sentence length and vocabulary richness. Syntactic features included frequency of function words and use of punctuation. Structural features included paragraph length and use of specific statements. Content-specific features included frequency of keywords. These features are extracted and learning algorithms are applied to three feature-based classification models, which are decision trees, back-propagation neural networks and Support Vector Machines (SVM). The experiment was based on Chinese messages. The accuracy fluctuates from 70% to 95% and SVM performed best as the classification technique.

Juola et al. (2006) presented an authorship attribution approach with a three-stage framework. The stages included pre-processing, comparison and ranking. Pre-processing techniques included lowercasing, punctuation removal and number replacement. The texts were then processed into non-overlapping words and their frequency distributions were compared against the standard distribution of the target texts.

As mentioned before, authorship identification was one of the three key tasks in the PAN'07 workshop. Five papers addressed the problem using stylometric features and classification algorithms. Stamatatos (2007) suggested treating the task as a profile-based text categorisation problem, which is different to previous instance-based approaches. A profile-based approach sorts all training texts by authors, whereas an instance-based approach treats each training instance separately. The profile-based approach was also adopted by Amitay et al. (2007) to identify multiple versions of a text by the same author.

Topic-independent features such as 2-grams of syntactic labels, vocabulary richness, the use of clauses, adverbials in sentences, sentence length, word length and character count are also used in authorship identification (Feiguina and Hirst, 2007; Karlgren and Eriksson, 2007; Mikros and Argiri, 2007). These features are also used in identification of translationese and translation direction research, which are described in the following section.

Statistical Language Models (Stolcke, 2002) are used in authorship attribution tasks. Language Models provide an efficient platform for n-gram comparison, and they are explored by Stein et al. (2008a); Stamatatos (2009). Stein et al. (2008a) also introduced meta analysis for authorship attribution, which includes a three-stage approach: a pre-analysis stage to determine what kind of model to apply in subsequent stages, a classification stage where writing styles are outlined, and a post-processing stage where the result of the previous stage is analysed with additional information. The first stage relies on features such as document length, genre, issuing organisation, and represents said features by language models. The second stage treats the document as a one-class classification problem. The final stage investigates additional information such as citation and uses a VSM to represent features for meta learning. This method helps to determine whether a set of texts is a subset of other texts.

Koppel and Schler (2004) presented their one-class machine learning classification approach to identifying whether a text is written by an author or not. A one-class task is where two texts are given and the goal is to identify whether the two texts were written by one author or by two different authors. Their approach investigates the level of difference between two texts, and the use of negative examples in the language model brought improvement to the classification. Koppel et al. (2009) proposed the use of a machine learning classifier. In three scenarios, that is, profiling challenge (no candidate set), needle-in-a-haystack challenge (many candidates) and authorship verification challenge (one suspect), they described how machine learning approaches can be adapted to various types of authorship attribution. The conclusion is that SVM and Bayesian regression are the most sophisticated solutions when used in conjunction with features such as character n-grams and function words/content words part-of-speech classes.

A survey of authorship attribution systems by Stamatatos (2009) summarises existing authorship attribution techniques as "inadequate" on their own. They claim that deep features such as syntactic and semantic information are only useful as a complement to other shallow features such as lexical information and n-grams. The argument is that the noise introduced by NLP tools during processing could contribute to their failure, which also applies to the use of NLP techniques for plagiarism detection. In addition, for plagiarism detection, the computational complexity is usually higher because of the number of pair-wise comparisons that need to be performed.

More recently, research has begun to investigate semantic information along with lexical and syntactic information in authorship attribution. Hedegaard and Simonsen (2011) investigated the use of semantic frames, using a frame-based classifier that provides word senses with annotated examples of their usage and meaning. FrameNet[11] is applied to identify authorship of translated and non-translated texts, and results showed that a combination of semantic frames and matching of frequent words performed better on translated texts, but matching of frequent words and n-grams performed better on non-translated texts.

A closely related field of research is **intrinsic plagiarism detection**, which refers to the detection of plagiarised passages within a document without the use of a source reference collection. Intrinsic plagiarism detection is very often related to authorship attribution. It looks for inconsistencies within a document by extracting lexical and syntactic features for each segment, and then compares the segments from the same document to find plagiarised segments that exhibits differences from the rest.

A study by Meyer zu Eissen and Stein (2006) discussed the use of statistical stylometric features which included statistics of text such as average sentence length, syntactic features that measure writing style at the sentence level, POS features to analyse the word classes, and frequency of special words based on tf-idf. Features were extracted from all segments of texts that belong to either the original or the plagiarised class. SVM is used for classification. The most promising features included average word frequency class, average number of prepositions

[11] https://framenet.icsi.berkeley.edu/fndrupal/

and average sentence length.

The study is followed up in Stein and Meyer zu Eissen (2007), where features from the previous research are investigated again. By evaluating the vocabulary richness, which is the ratio dividing the number of unique word types by the number of tokens, of each passage within a document, analysis can be performed at passage level. The experiment was performed on a corpus of 50 German documents with manually plagiarised texts. POS of words and statistical features such as average sentence length, frequency classes of words such as adjectives and verbs, average word length and average number of stopwords were used in a linear discriminant classifier. The results showed a combination of features performed better than individual features.

3.4.2 Cross-lingual plagiarism detection

Cross-lingual plagiarism detection refers to plagiarism cases where the language in the source texts is not the same as the language in the plagiarised texts. This can be done by using translation software or manual translation to convert the language of the source texts. A pair of texts is considered to be plagiarised if they are semantically similar, regardless of the language difference. Cross-lingual plagiarism detection is a complex task. Unlike monolingual plagiarism detection which can be addressed by word overlap measures, cross-lingual plagiarism cases cannot be directly compared word-for-word. Before making a comparison, the following questions must be answered:

1. How do we determine the source language of a translated text in the plagia-

rised document?

2. Should the comparison be done on one language only and which language should it be?

To help address these questions, techniques that determine translation direction can be used, where the source language of a piece of translated text can be identified. For example, in Baroni and Bernardini (2006) experiments were performed on a domain-specific corpus consisting of English, Arabic, French, Spanish and Russian texts translated into Italian. The experiment was performed using an SVM classifier, based on features such as lemmatised words and POS sequences. The best accuracy was achieved by using a combination of features that includes 1-gram word with tf-idf weighting, and 2-grams and 3-grams of POS tags. The experiment concluded that the task relies on the distribution of n-grams of function words and morpho-syntactic features.

Pouliquen et al. (2003) presented a statistical approach to map multilingual documents to a language-independent document representation, which measures similarity between monolingual and cross-lingual documents. A parallel corpus with multilingual translated texts was used, and pre-processing techniques including lemmatisation and stopword removal were applied. Parallel texts in different languages are identified by the tf-idf of the topic, and the top 100 words are selected as “descriptors”. Each descriptor contains one-to-one translations into different languages and is represented by a vector. The similarity score was calculated by comparing the vectors between Spanish and English documents. The approach used in this study is not based on n-gram comparison, which is a language-specific technique that is limited to monolingual plagiarism. The proposed approach may be potentially suitable for detecting cross-lingual plagiarism, and further work is suggested to be performed on paraphrased translations.

To facilitate direct comparisons between multilingual texts, source texts in the collection can be translated to the language of the suspicious texts (or vice-versa) by manual or mechanical means. This allows comparison to be based on standard monolingual approaches. If the plagiarised text was translated manually from source text, involving human translators may not be feasible if the document collection is large, but mechanical means do not always provide accurate translations comparable with manual translations.

However, if the source text is translated via mechanical means then the identification is much simpler. Machine translated cases can easily be identified with a reverse-translation approach, particularly if the same machine translation system is used. For instance, in the PAN competition translated plagiarism cases are processed with a machine translation approach. It is a common approach for competitors to first use a language identification toolkit to identify the source language of non-English documents, and then use a machine translation toolkit to translate all non-English documents to English before processing all documents in English. Barrón-Cedeño and Rosso (2010) posit that this approach is effective because the corpus was created using a similar method.

Statistical methods for cross-lingual plagiarism detection have been applied in Barrón-Cedeño et al. (2008). They described the use of the IBM Model 1 alignment model with a statistical bilingual dictionary to analyse plagiarism in a parallel corpus. Preliminary experiments on English and Spanish text fragments achieved satisfactory results, but further experiments are needed on a cross-lingual corpus in order to evaluate the approach. The extension of the previous work (Pinto et al., 2009) was tested on English versus Spanish, and English versus Italian documents. The approach is again performed using the IBM Model 1 alignment model based on a bilingual statistical dictionary, which directly captures correlated words across languages. These studies suggested that alignment could be beneficial to other cross-lingual information retrieval tasks.

Potthast et al. (2010a) performed a cross-lingual plagiarism detection experiment on a large-scale multilingual corpus. The six languages tested are English, German, Spanish, French, Dutch and Polish. The three-stage framework included heuristic retrieval, detailed analysis, and knowledge-based post-processing. In the first stage, features were extracted using keyword extraction and fingerprints of documents were generated. The second stage involved information retrieval using VSM, cross-language alignment-based character n-gram model, and statistical bilingual translation model. The final stage aimed to reduce the number of false positives by checking if the flagged cases have been cited or not. The results showed that detection performance is heavily based on either the syntactical relatedness of the languages or the accuracy of translations. They also pointed out that the alignment model achieved good results on automatic translations and it can also be applied to language pairs with low syntactic relatedness.

3.4.3 Other text similarity approaches

Other text reuse detection includes near duplicate detection, which is a task that identifies documents that are nearly identical. The difference between near duplicate detection and plagiarism detection is that the first refers to many documents having one reference. The latter refers to suspicious documents which can have more than one reference copy and plagiarism is not exclusive to document level copy.

NLP has been applied to the detection of duplicated technical reports , as described in the work by Runeson et al. (2007). They used natural language processing including tokenisation, stemming, stopword removal, and synonym replacement. The words were then represented in a VSM, and similarity between two texts was measured in the vector space. Although the techniques used are simple and the experimental setting is not extensive, the experiment shed light on the incorporation of NLP techniques to support the identification of similar texts.

Yang and Callan (2006) introduced a clustering approach to near duplicate detection. Documents were split into blocks and the level of changes in each block was measured using features such as the similarity between bag-of-words, and the edit-distance between words in blocks. They suggest that clustering is a more effective way to handle large corpus and further advocate combining features that include textual and non-textual information.

Manku et al. (2007) proposed a near duplicate detection method for web documents. Their method uses small-scale fingerprinting, which is the representation of features, including processed texts by tokenisation, stop-word removal and stemming, in a document using a sequence of vectors. This is used in conjunction with the Jaccard coefficient (Formula 4.1 on page 103) to measure the level of similarity. Their approach aims to handle web data, which was the reason why a small-scale fingerprint must be used. The initial experiment on a small set of web data highlighted further challenges, such as the variation of document length, categorisation of multiple languages, and the tuning of the sensitivity of detection algorithm.

Xiao et al. (2008) proposed the integration of new filtering algorithms with an existing near duplicate detection approach. Positional filtering that exploits the order of the word tokens is combined with existing overlapping similarity measures. The similarity is calculated by the Jaccard and cosine metrics. They conclude that combining the approaches can help to improve the efficiency and effectiveness of filtering duplicate documents.

Compared to VSM, the Normalised Word Vectors algorithm allows a larger dimensional space to represent the content of the document. This is applied to automatically grade essays and to classify documents in digital libraries to detect similar texts (Dreher, 2007; Williams, 2006; Parker et al., 2008).

Another related area of research is the detection of journalism text reuse. The work by Clough et al. (2002a) shows three approaches to distinguishing originals from derived newswire texts. Their methodology involves a supervised machine learning model that includes three features: 3-gram overlap measure, Greedy-String-Tiling and sentence alignment. The task is treated at the document level and the results show that a combination of the features yields better results than the features on their own. The difference between plagiarism detection and journalism text reuse is that in the latter one piece of original text can result in many other derived texts, whereas in the study of plagiarism detection it can go both ways: a plagiarised text can come from more than one original text, and one original text can attribute to many plagiarised texts. Furthermore, the principles of the two fields are different. The Press Association is the main news source provider and their newswire service is used by many papers in the UK. The original news source is distributed to individual news agencies, which then re-word the news and publish it to the public. As it is not the intention of the journalists to "plagiarise" pieces of news, they do not try to conceal the fact the texts are not original. Therefore, journalism text reuse should not be treated as plagiarism per se, as the nature of rewriting is different.

Similarly, Tashiro et al. (2007) developed a simple approach for detecting copyright infringement texts from the web. The similarity between texts was calculated based on the longest common subsequence of 2-grams of words. A threshold is then set to determine whether a pair of texts was similar or not. The experiment was performed on news and lyrics with short texts of a length between 163 and 788 words. Although the results showed a 94% precision, the reality is that longest common subsequence is not ideal for detecting similarity in longer texts lengths as the computational cost is high.

The degree of semantic equivalence is explored in Semantic Textual Similarity (STS) tasks. The level of similarity between two sentences is measured by analysing the semantic components. Studies in the STS task is related to paraphrase and textual entailment, which is described later. The difference is that a textual entailment task is directional, whereas a STS task is bidirectional. The outcome of a STS task is not a binary classificaiton but rather it assigns a level of similarity between sentences. The pilot STS task in 2012 introduced a training and testing corpus which contained sentence pairs from paraphrase datasets, machine translation evaluation datasets, and lexical resource mapping exercise (Agirre et al., 2012). The similarity of sentence pairs is rated on a scale of 0-5, with 0 being the least similar and 5 the most similar. The results from 35 teams are compared with human judgement from Mechanical Turk with a Pearson correlation of 90%, and the best team scored about 80%. The techniques and resources used in the task include synonym generalisation using WordNet, stopword removal, paraphrase matching, lemmatisation, POS tagging and semantic role labeling.

The winning approach in the STS 2012 task by Bär et al. (2012) combined simple features such as n-grams of characters, words, POS tags and stopwords, and complex features such as pairwise word similarity by means of calculating the idf-weighted best-matching words in both directions. Lexical-semantic resources which include WordNet and Wikipedia are used as part of the semantic analysis, where word sense disambiguation is applied in the noun substitution stage. The features are combined to compute similarity scores in a machine learning linear regression classifier with a 10-fold cross-validation.

Androutsopoulos and Malakasiotis (2009) provided a comprehensive survey of NLP techniques in detecting paraphrases and textual entailment tasks. The survey concluded that existing approaches exploit a combination of superficial features such as surface string comparison, shallow semantics such as verb generalisation using VerbNet, semantic role labelling using PropBank, and deeper syntactic features such as dependency trees representation. Alignment techniques from statistical machine translation that can exploit large bilingual parallel corpora are also applied in RTE tasks. This work can be further explored in the plagiarism detection field.

To explore paraphrases within texts, a field of interest is the Recognising Textual Entailment (RTE) task. It focuses on textual inference and very often refers to semantic variations between pairs of expressions. Given a text **T** (multiple sentences) and a hypothesis **H** (single sentence), the aim of RTE is to detect whether **T** is inferred from **H**. The major difference between paraphrasing and textual entailment is that paraphrasing is bi-directional, whereas textual entailment only infers that a text **T** entails a hypothesis **H**. Starting in 2006, the RTE challenge has now progressed to the eighth edition, attracting substantial interest (Dagan et al., 2006).

Simply put, textual entailment is a sentence level paraphrase that can include other semantic variations. This is different from paraphrasing, which normally consists of expressions of equal length and the T and H are inferred bi-directionally. The difference can be illustrated with an example of textual entailment (**H** entails **T**):

Text T: Medical science indicates increased risks of tumours, cancer, genetic damage and other health problems from the use of cell phones.

Hypothesis H: Cell phones pose health risks.

From the example, one can see that textual entailment is a case where the meaning of sentences is uni-directional, which means that **T** entails **H** but **H** does not necessary entails **T**.

An example of paraphrasing, or bi-directional entailment (where **T** entails **H**, and **H** entails **T**) would be the following:

Text T: Although humans are comparatively poor sprinters, they also engage in a different type of running, such as endurance running, defined as running many kilometres over extended time periods using aerobic metabolism.

Hypothesis H: Having limited success in sprinting compared to other mammals, humans perform better in endurance running, which is a form of aerobic running over extended distances and periods of time.

This example shows that paraphrasing is a type of textual entailment, but the entailment is bi-directional, which means that **T** entails **H** and vice-versa.

Methods used in textual entailment may help work on plagiarism detection, but they are not designed to accommodate the processing of document level pairwise comparisons, neither the processing of very large collections of texts such as those used for plagiarism detection, particularly deeper RTE approaches.

One of the available systems that tackles the RTE task is VENSES (Delmonte et al., 2005), which performs semantic evaluation for textual entailment. It exploits both shallow and deep linguistic features of texts, using techniques including lexical generalisation with disambiguation, dependency relation matching, named entity recognition, POS tagging based on finite state automata, and semantic role processing. It provides detailed output of grammatical relations and semantic role labels. As the textual entailment task is performed on short texts, an initial small-scale experiment of plagiarism detection using VENSES is described in Chapter 5.

To measure the similarity between texts based on semantics, Corley and Mihalcea (2005) introduced the use of a WordNet-based similarity metric. The metric pairs up words that are similar and weights are given for each pair of words. The metric is combined with language models, and the best performance was achieved by combination of this similarity metric and standard lexical matching. Their research shows improvement in directional entailment tasks, reaching an accuracy of around 58%. They concluded that the method disregards many relations in the sentence structure, as well as the arguments and the dependencies between words, and a more sophisticated approach is needed to process this deeper linguistic information.

Another related area is the detection of multilingual paraphrases. Zhao et al. (2009) extracted English paraphrases from a bilingual English-Chinese parallel corpus, and performed an experiment on one million paraphrases. Their proposed method focused on maximum likelihood estimation of paraphrases, lexical weighting and monolingual word alignment. Their paraphrases were classified as five types, including 1) trivial changes such as inserting/deleting stopwords, 2) phrase replacement that replaces words but retains their POS order, 3) phrase reordering in which words are reordered within a sentence, 4) structural paraphrases in which the words are significantly changed but the meaning is the same, and 5) information addition/deletion where words are added or deleted from the sentence but it maintains the same meaning. Out of the five types, only 4 and 5 are considered as complex paraphrasing that poses a challenge in other NLP tasks.

3.4.4 Plagiarism direction detection

Current research in plagiarism detection is mostly focused on the detection of plagiarised texts within a document collection or within a document, and the direction of plagiarism is predetermined. Source documents and suspicious documents are provided separately and the task is to determine which segments of the suspicious text are copied from which segments of source texts. The similarity score is given by a pair-wise comparison metric using features such as word overlap. The limitation of this approach is that if the document collection is large, a large number of pair-wise comparisons will be required to perform filtering and detection. Besides, in a real-world scenario, it is often difficult to determine whether a piece of text is the original or another plagiarised version.

This is the problem faced by online commercial plagiarism detection products: it is not uncommon to find cases where a "plagiarised text" is actually the original text, as a plagiarised version was submitted before the original. After all, a plagiarism detection tool can only suggest there are similarities between two pieces of texts, but it cannot determine the plagiarism direction.

Thus, we also investigate a novel perspective on plagiarism research in this thesis: instead of measuring the similarity between pairs of texts, the task is to distinguish source text from plagiarised text. This is achieved by investigating the linguistic and statistical traits presented in the document collection, finding a pattern for the two types of texts (source and plagiarised), and classifying each individual text into its respective group.

To date, research on the detection of plagiarism direction is very limited. Grozea and Popescu (2010) applied their plagiarism detection system Encoplot, which is based on character 16-grams comparison, to artificially-generated plagiarised documents from the PAN corpus. The cases are generated via automatic means with various obfuscation levels, and the results showed that at the document level the overall accuracy can reach approximately 75%. The tests on highly obfuscated artificial documents reached an accuracy of 69.77%. Analysis of the research shows there are significant and measurable differences between original and plagiarised texts in the PAN corpus. To the best of author's knowledge, no research has been done on manually plagiarised documents and at the passage level.

An interesting study by Ryu et al. (2008) proposed an algorithm to measure the direction of plagiarism, in other words, to determine which is the suspicious document or the original document. The new algorithm is based on the distance measure evolutionary distance. However, the research is language-dependent and it is limited to Korean.

Hence, in Chong and Specia (2012) we proposed a framework to distinguish plagiarised from original texts by using linguistic and statistical traits of texts. The framework was tested in two tasks: 1) the classification of individual text segments as original or rewritten, and 2) the ranking of two versions of a text segment according to their originality to determine the rewriting direction. The approach does not involve comparison between many suspicious texts and source texts, but focuses on building a pattern of rewriting traits and fitting each text segment into their classes. A detailed description of this experiment is presented in Chapter 7.

Statistical features can be generated by language models, similar to the experiment performed by Lavergne et al. (2008) which distinguishes natural texts from artificially generated ones, described in Section 3.1.3. On the other hand, linguistic features are inspired by the studies on translation direction and translationese, which aim to distinguish original and translated texts, based on the ontological differences between those texts. It follows Translation Universal theory (Gellerstam, 1986), which posits that a few universal principles apply when humans perform translations, regardless of the languages involved. One such a principle is that of simplification. The simplification hypothesis states that translated texts tend to be simpler than the original and that translated texts are likely to keep specific properties which can be identified via lexical, grammatical and syntactical means (Baker, 1993, 1996). In our study, it is not the aim to apply translation universal theory directly to plagiarism direction, but we seek insights from this field that may be beneficial to the task. This study also considers features that are inspired by the simplification universal. It is important to note that although the corpus used will consist of monolingual English texts, studies in translationese are also tested on monolingual comparable corpora.

Nahnsen et al. (2005) proposed a method to identify multiple versions of translated texts from one original text. The experiment was performed on book chapters with multiple parallel translations. 4-grams of lexical items were extracted, which included nouns, verbs and adjectives from the texts. To generalise each lexical item, each word in the sentence was disambiguated using WSD to determine the most suitable sense of the word. Similarity was determined by cosine similarity on n-grams of lexical items and tf-idf on weighted keywords. The results show that n-grams of lexical items, based on shallow semantics by lexical generalisation, can outperform traditional statistical methods.

To identify the difference between original and translated texts, Baroni and Bernardini (2006) use tf-idf of 1-grams, 2-grams and 3-grams of word, POS tags and lemmas and classify them using SVM. The corpus consists of monolingual texts, as both the original non-translated and the target translated texts are in Italian. The results show that the most promising features include the distribution of function words, personal pronouns and adverbs. The study was followed by Kurokawa et al. (2009), and their experiment reported an accuracy of 90% on n-grams and 77% on sentences when detecting the direction of translation. Their experiment was performed on an English-French parallel corpus, with features such as POS and lemmas in an SVM machine learning classifier. They tested up to 5-grams and the best accuracy was achieved using 2-grams of words.

A study of six languages by Halteren (2008) using frequency counts of word n-grams shows that it is possible to distinguish between translated and non-translated texts and to identify their respective original languages. This is followed by the work of Lembersky et al. (2011, 2012) which focused on building statistical language models for each language to aid the identification of translated texts. They show that translated texts from different original languages display sufficient traits that can be identified.

In addition to using language models, the studies by Ilisei and Inkpen (2011); Ilisei et al. (2010) on Romanian and Spanish translationese describe a machine learning approach that use morphological and simplification features. Their studies showed that the highest contributing simplification features are information load and lexical richness, and the best performing morphological features include the proportion of nouns, pronouns and finite verbs over tokens.

A related study by Volansky et al. (2011) explores the differences between original, manually translated and machine translated texts. Linguistically-motivated features that include simplification features are employed in an SVM classifier.

These experiments on translation direction confirm that translated texts have lower lexical richness and higher numbers of frequent words. It is pointed out that simplification features alone are not sufficient to distinguish between original and translated texts, but they help to improve the accuracy when combined with other features.

As these studies in translationese and translation direction detection have suggested that shallow data representations are applicable in the classification of translated and non-translated texts, a language-independent model based on simplification, morphological, syntactic and statistical features is proposed and investigated in this thesis. The plagiarism direction identification framework is described in detail in Chapter 7.

3.5 Challenges in Plagiarism Detection

This section describes challenges faced by existing plagiarism detection approaches. These challenges can be grouped into two main areas: linguistic complexity and technical difficulty.

First, overlap word n-grams may be very effective against word-for-word copies, but plagiarism cases are more complex than verbatim copy-and-paste. The three main linguistic challenges are 1) lexical changes, 2) structural changes, and 3) paraphrases.

1. **Lexical changes.** This refers to the use of synonymy or related concepts to replace original words, which is essentially having two words carrying the same meaning but with different representations. For example (texts excerpted from the PAN-PC-10 corpus):

   Source: When this man returned he brought me a letter from your father, in which he said he was going to try and make his escape, and that he would never again set foot in Russia.

   Lexical change: When this man returned he conveyed me a note from your dad, in which he said he was going to trial and make his get away, and that he would not ever afresh set base in Russia.

2. **Structural changes.** This refers to the modification of active/passive voice, changes in word order, re-ordering of sentence components while maintaining

the original meaning.

Source: Even Beckwith, who could not coincide with others as to the great importance of intemperance as an etiological element, says distinctly, that intemperance was, by far, the most potent of all removable causes of mental disease.

Structural change: Even Beckwith, who didn't agree that intemperance was important as an etiological element, said that intemperance was the strongest of all removable causes of mental illness.

3. **Paraphrases.** This refers to the most complex form of text operations and combines lexical and structural changes. The text is represented using different words and structures, and possibly with different lengths, but the meaning remains the same.

   Source: I have heard many accounts of him, said Emily, all differing from each other: I think, however, that the generality of people rather incline to Mrs. Dalton's opinion than to yours, Lady Margaret. I can easily believe it.

   Paraphrase: Emily said, I have heard many different things about him; however, most people trust Mrs. Dalton's beliefs more then they do yours, Lady Margaret, myself included.

Overlapping n-gram alone is insufficient to identify similarity between these text pairs, but with the use of lexical generalisation, it is possible to recognise synonyms and deal with lexical changes. Syntactic and semantic parsing can help to identify the structure of texts, while other levels of processing such as named entity recognition can highlight important concepts in the texts. Our hypothesis is that these techniques and other NLP techniques can help identifying complex cases of plagiarism, but they have issues and challenges of their own, as we later discuss in this thesis.

Technical difficulties also limit system performance. The main constraint is computational resources. To begin with, performing pair-wise comparisons in large document collections requires significant processing and memory resources. This is especially problematic as plagiarism can be derived from multiple sources. A plagiarised document may contain text segments from more than one source, and it is difficult to identify the possible source segments if the initial document level pair-wise comparison failed to establish the candidate documents. In other words, detecting plagiarism from multiple sources is more difficult than from a single source, as some detection metrics only relate a suspicious document to one source document.

Moreover, the difficulty of obtaining a real-life corpus means that experiments are limited to using specially-created corpora. Although such corpora contain some manually rewritten texts, some of them are plagiarism cases generated via artificial means, which adds an extra challenge to the application of NLP techniques, as artificially generated cases are not linguistically well-structured and therefore existing tools cannot reliably process them.

These are some of the challenges which plagiarism detection approaches face today. Using only string-matching will not be sufficient to tackle these issues effectively. Additionally, the use of complicated methods require vast amounts of computational resources (Bao et al., 2004). The trade-off between computational speed and detection reliability needs to be considered when applying algorithms.

In this thesis, the linguistic challenges that will be addressed are lexical changes, structural changes and paraphrasing. These are the challenges that motivated the use of NLP in plagiarism detection. The technical challenges are alleviated by having a filtering step based on simple processing, as typical of plagiarism detection approach, but overall our focus is not on addressing this type of challenge.

3.6 Summary

This chapter described the existing approaches to plagiarism detection. This helps to meet the **first objective** by conducting a thorough investigation of current techniques and approaches, thereby providing a fundamental understanding for proposing a plagiarism detection framework in the following chapter. In this chapter, it is noted that most existing methods are based on brute-force string-matching algorithms, and the use of NLP techniques in plagiarism detection is underexplored. The existing methods follow a three-stage detection approach which will be incorporated into the proposed framework. The chapter also described other related research on intrinsic plagiarism detection, cross-lingual plagiarism detection, other text similarity detection and translation direction detection that provided inspiration for the proposed methodology.

Chapter 4

A Framework for Natural Language Processing in Plagiarism Detection

This chapter describes our framework for the incorporation of simple text preprocessing, and shallow and deep NLP techniques in automatic plagiarism detection.

Section 4.1 presents the proposed framework which is used throughout the thesis. The framework applies a broad range of NLP techniques which aim to improve the performance of plagiarism detection. Section 4.2 describes the text preprocessing techniques, shallow NLP techniques and deep NLP techniques. Section 4.3 lists the metrics for similarity comparisons between texts. Section 4.4 details the machine learning algorithms for text classification. The chapter concludes with Section 4.5, which describes the evaluation metrics used.

4.1 General Framework

The framework for external plagiarism detection involves five stages. It is an expansion of the the common three-stage approach described in Section 3.1.

Stage 1: Pre-processing This stage prepares the input text collection, including both suspicious and source texts, for subsequent stages. Text pre-processing and shallow NLP techniques are applied to the texts.

Stage 2: Similarity comparison This stage performs pair-wise comparisons between each suspicious text against all source texts. One or more similarity metrics are applied to give each suspicious-source text pair a similarity score.

Stage 3: Filtering The similarity scores generated in Stage 2 are used to judge the likelihood of a suspicious-source pair being listed as a candidate pair. The likelihood is determined by setting a threshold on the similarity scores. This can be done either by using a machine learning algorithm to learn the threshold, or by manually defining such a threshold. If a pair has reached a certain threshold, the pair is listed as a candidate pair; otherwise the pair is discarded as not plagiarised.

Stage 4: Further processing As deep linguistic features are computationally expensive, this stage is only applied to candidate pairs. Candidate pairs from Stage 3 are further processed; then Stage 2 is repeated for the pairs of Stage 4 to generate a similarity score.

Stage 5: Classification The final stage is to use the similarity scores from the previous stage to assign each text pair a classification as *Plagiarised* or *Clean*. In some cases the class Plagiarised can be further defined at various levels, such as Near Copy, Heavy Revision, or Light Revision. The classification is either done by setting thresholds, or by using similarity scores generated from various modules as features in a machine learning classifier. Classifications are verified by applying standard evaluation metrics which include precision, recall, f-score and accuracy.

The processing flow chart (Figure 4.1) shows the general framework proposed in this study. A text collection pass through various stages of processing, and then similarity metrics are applied to compute the similarity between texts for each suspicious-source pair. The similarity scores resulting from shallow techniques are used as features in text classification or in the filtering stage before applying deep NLP techniques. This five-stage framework has been applied in the small-scale experiment described in Chapter 5 and the large-scale experiment described in Chapter 6.

4.2 Text Pre-processing and Natural Language Processing Techniques

This section describes the text pre-processing techniques (Section 4.2.1), the shallow NLP techniques (Section 4.2.1), and the deep NLP techniques in (Section 4.2.3) used in our experiments.

4.2.1 Text pre-processing techniques

These techniques are available from the Python module of the Natural Language Processing Toolkit[12] (NLTK), which aids text analysis and development. The techniques used are as follows (example texts excerpted from the PAN-PC-10 corpus):

Sentence segmentation This technique splits the text in the document into sentences, which allows sentence-by-sentence processing in the subsequent

[12] http://nltk.org/

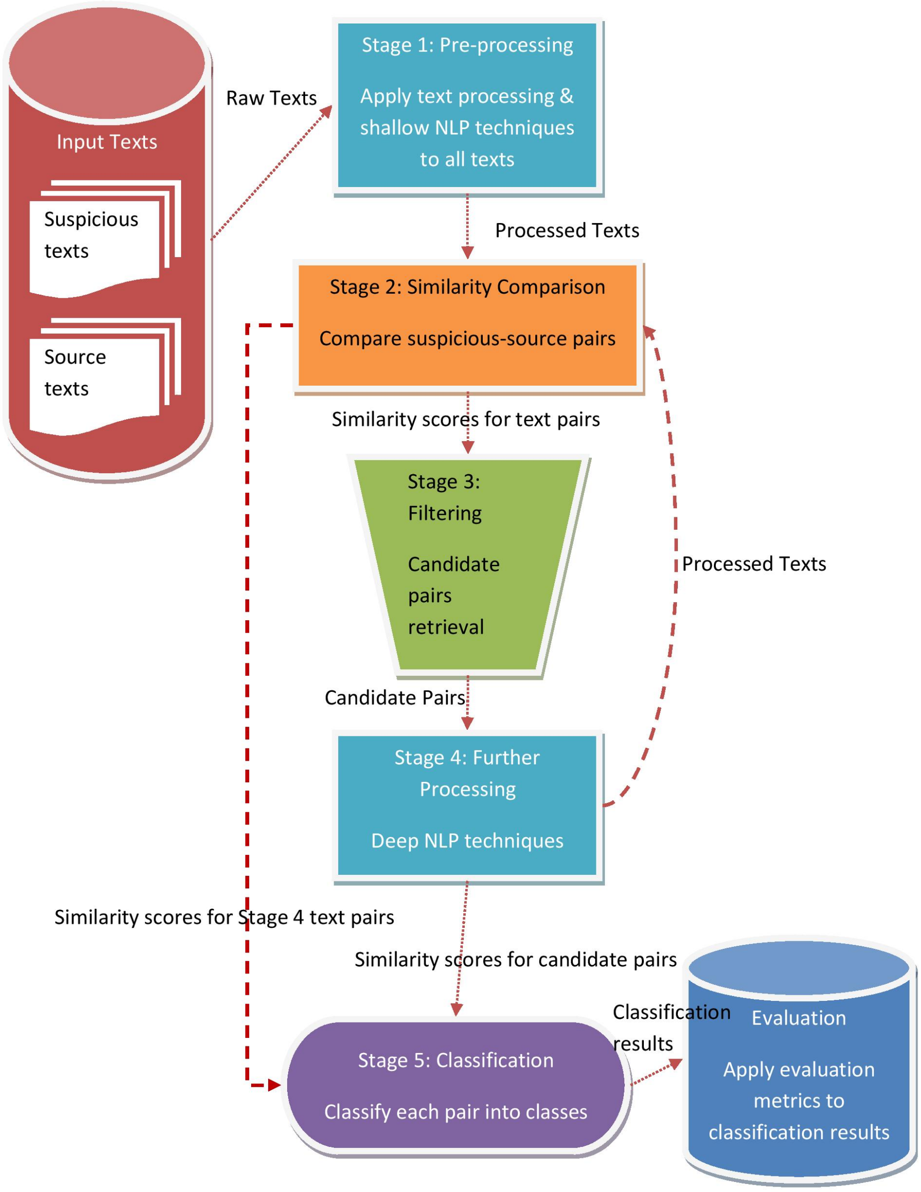
Input Texts
Suspicious texts
Source texts
Raw Texts
Stage 1: Pre-processing
Apply text processing & shallow NLP techniques to all texts
Processed Texts
Stage 2: Similarity Comparison
Compare suspicious-source pairs
Similarity scores for text pairs
Stage 3: Filtering
Candidate pairs retrieval
Processed Texts
Candidate Pairs
Stage 4: Further Processing
Deep NLP techniques
Similarity scores for Stage 4 text pairs
Similarity scores for candidate pairs
Classification results
Evaluation
Apply evaluation metrics to classification results
Stage 5: Classification
Classify each pair into classes

Figure 4.1: External plagiarism detection framework

stages. For example:

Raw text: Therefore, a person should search his actions and repent his transgressions previous to the day of judgment. In the month of Elul

(September) he should arouse himself to a consciousness of the dread justice awaiting all mankind.

Sentence segmentation: (Sentence 1) (Therefore, a person should search his actions and repent his transgressions previous to the day of judgment.) (Sentence 2) (In the month of Elul (September) he should arouse himself to a consciousness of the dread justice awaiting all mankind.)

Tokenisation This technique determines token boundaries, such as words and punctuation symbols in sentences. For example:

Raw text: Therefore, a person should search his actions and repent his transgressions previous to the day of judgment.

Tokenisation: (Token 1, Token 2, Token 3... Token n) (Token 1 = “Therefore” Token 2 = “,” Token 3 = “a”... Token 18 = “judgment” Token 19 = “.”)

Lowercasing This technique substitutes every uppercase letter with lowercase to generalise the matching. Using the same example from above:

Raw text: Therefore, a person should search his actions and repent his transgressions previous to the day of judgment.

Lowercase: therefore, a person should search his actions and repent his transgressions previous to the day of judgment.

Stopword removal This technique removes function words, which include articles, pronouns, prepositions, complementisers, and determiners, such as “the”, “of”, “a”, “and”. Using the same example from above:

Raw text: Therefore, a person should search his actions and repent his transgressions previous to the day of judgment.

Stopword removal: Therefore, ~~a~~ person should search his actions ~~and~~ repent his transgressions previous ~~to the~~ day ~~of~~ judgment.

Punctuation removal This technique removes punctuation symbols to generalise matching between tokens. Using the same example from above:

Raw text: Therefore, a person should search his actions and repent his transgressions previous to the day of judgment.

Punctuation removal: Therefore a person should search his actions and repent his transgressions previous to the day of judgment

Number replacement This technique replaces numbers and figures with a dummy symbol in order to generalise the texts for matching. For example:

Raw text: Without enumerating all the modern authors who hold this view, we will quote a work which has just appeared with the imprimatur of Father Lepidi, the Master of the Sacred Palace, in which we find the two following theses proved: 1.

Number replacement: Without enumerating all the modern authors who hold this view, we will quote a work which has just appeared with the imprimatur of Father Lepidi, the Master of the Sacred Palace, in which we find the two following theses proved: [NUM].

Text pre-processing techniques are normally applied as a combination. To illustrate the process, given the example of the following raw texts:

Source (s): When this man returned he brought me a letter from your father, in which he said he was going to try and make his escape, and that he would never again set foot in Russia.

Plagiarised (p): When this man returned he conveyed me a note from your dad, in which he said he was going to trial and make his get away, and that he would not ever afresh set base in Russia.

The following outputs are produced after applying tokenisation, lowercasing, punctuation removal and stopword removal:

(s) man returned brought letter father going try make escape never set foot russia

(p) man returned conveyed note dad going trial make get away ever afresh set base russia

4.2.2 Shallow NLP techniques

Shallow NLP techniques help to analyse the morphological traits of texts, and they do not provide syntactic and semantic analysis of the text. These techniques are available from the NLTK toolkit or the Stanford CoreNLP toolkit.[13]

Part-of-speech tagging This technique assigns grammatical tags to each word, such as "noun", "verb", etc., for detecting cases where words are replaced, but the style in terms of grammatical categories remains similar.

[13] http://nlp.stanford.edu/software/corenlp.shtml

Raw text: (text s) Set foot in Russia (text p) Set base in Russia

POS-tagging:

(s) Set [VBN] foot [NN] in [IN] Russia [NNP]

(p) Set [VBN] base [NN] in [IN] Russia [NNP]

Lemmatisation This technique transforms words into their dictionary base forms, which generalises the texts for similarity analysis. For example, "produce" and "produced" are normalised to "produce".

Stemming This technique transforms words into their stems, which generalises the texts for similarity analysis. For example, both "computer" and "computers" are normalised to "comput", and "product", "produce", and "produced" to "produc".

Chunking This technique is also called shallow parsing. It identifies the phrasal constituents in a sentence, including noun phrase, verbal phrase, etc., and splits the sentence into chunks of semantically related words. It is a shallow NLP technique as it does not specify the internal structure or the role of words in the sentence. Chunking can provide a relatively less computationally-expensive solution for analysing the structure of texts.

Raw text: When this man returned he brought me a letter from your father, in which he said he was going to try and make his escape, and that he would never again set foot in Russia.

Chunks: [ADVP When] [NP this man] [VP returned] [NP he] [VP brought] [NP me] [NP a letter] [PP from] [NP your father] , [PP in] [NP which]

[NP he] [VP said] [NP he] [VP was going to try and make] [NP his escape] , and [SBAR that] [NP he] [VP would never again set] [NP foot] [PP in] [NP Russia] .

4.2.3 Deep NLP Techniques

This section describes the deep NLP techniques which help to analyse the syntactic and semantic traits of texts. As superficial techniques are not sufficient to identify complex plagiarism cases that involve paraphrases, deep techniques which are not dependent on word-for-word comparison can provide another perspective for text analysis.

Dependency relation extraction This technique of deep syntactic analysis returns, for a given sentence, the syntactic relationship between each pair of words. Before applying parsing to the texts, sentence segmentation is applied to determine the sentence boundaries. The Stanford Parser[14] version 1.6.5 (de Marneffe et al., 2006) is then applied to generate output in the form of dependency relations, which represent the syntactic relations within each sentence. This allows the similarity comparison to be based on the syntactic relations between words, instead of having to match words in their exact order in n-gram based comparisons. For example, for the sentence “A basic concept of Object-Oriented Programming.”, the following relations are produced:

```
det(concept-3, A-1)
```

[14] http://nlp.stanford.edu/software/lex-parser.shtml

```
amod(concept-3, basic-2)
prep(concept-3, of-4)
nn(Programming-6, Object-Oriented-5)
pobj(of-4, Programming-6)
```

Syntactic constituent extraction Another deep NLP technique is the analysis of syntactic constituents for each sentence. The tool VENSES[15] (Delmonte et al., 2005), which is a Recognising Textual Entailment (RTE) tool, provides analysis on sentence level syntactic constituents. Instead of using the tool in an entailment task, we extract the syntactic constituents from the analytical output of the framework. This allows the similarity comparison to be done at a syntactic level, which is not limited to n-gram matching of exact words. For example, for the sentence "Inheritance is a basic concept in object-oriented programming." the following output is extracted:

```
Syntactic constituents:
subj-[Inheritance-n-sn]
ibar-[ (is)-ause-ibar]
xcomp-[a-art-sn, basic-ag-sn, concept-n-sn]
obl-[in-par-_G36673, object_oriented-vin-ibar, programming-n-sn]
```

Lexical generalisation Generalising words for word-level matching is not completely new in plagiarism detection approaches. However, most approaches face the problem of Word Sense Disambiguation (WSD), which means each word needs to be disambiguated to find an appropriate meaning and a related synonym. WSD is a difficult task on its own, which in turn affects the synonym generalisation progress. Hence, we propose to retrieve and compare all groups of synonyms of a word in all its senses, making it possible to

[15] http://project.cgm.unive.it/venses_en.html

achieve a matching even if the plagiarised word has been substituted with another word of similar meaning. This approach was described in Chong et al. (2010); Chong and Specia (2011) where all synsets were selected. Synonyms are retrieved from the WordNet[16] lexical database, which provides a hierarchical representation of synsets, that is, conceptually related groups of synonym words.

For the experiments with lexical generalisation, function words **(stopwords)** are removed and all remaining (content) words are generalised using WordNet. WordNet **lemmatises** words and generates synsets for each content word. In other words, this technique expands the source and suspicious texts by replacing each content word by the words (synonyns) in all of its synsets from WordNet. For each word in the source and suspicious documents, all the synsets are extracted. Word ambiguity is not a problem in this case as all synsets will be selected regardless of the context, and therefore it is not necessary to apply WSD techniques.

[16] http://wordnet.princeton.edu/

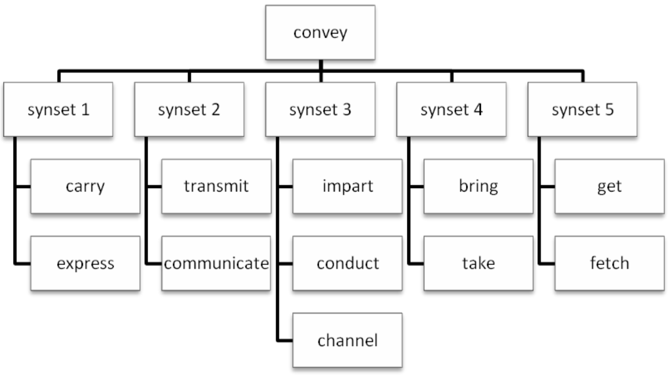

Figure 4.2: Example of synsets of the word "convey"

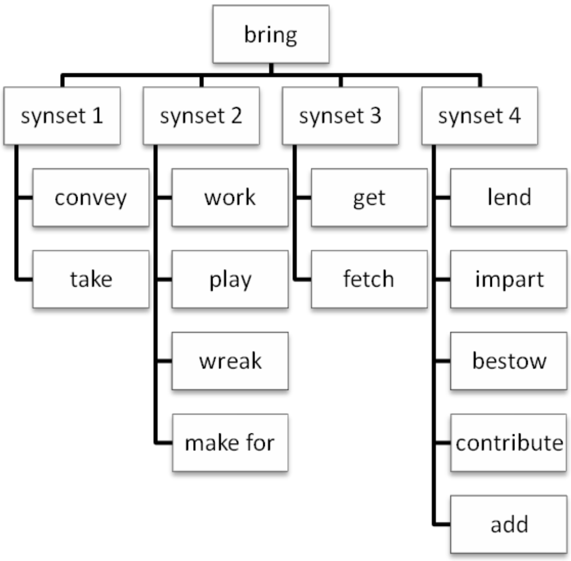

Figure 4.3: Example of synsets of the word "bring"

In Figures 4.2 and 4.3 the synsets of the words "convey" and "bring" are extracted. Although the words carry the same meaning, word-for-word matching metrics will not identify them as similar. By comparing the synsets of words, we can see that "convey" matches synset 1 of "bring", and "bring" matches synset 4 of “convey”.

Predicate generalisation To analyse the grammatical components of a sentence, we propose to analyse the predicate of a sentence, which can be represented by the verbs within it. Hence in this technique verbs are extracted from texts, and generalised for similarity comparison. We use NLTK to transform each verb to lowercase and then apply the WordNet lemmatiser module before looking up the verb class in VerbNet[17]. VerbNet provides lexical resources that organise verbs into hierarchical classes. Each verb is represented by its respective VerbClass which contains other sub-classes that are syntactically or semantically related to other verbs of the same class. To generalise verbs, each verb in source and suspicious texts is replaced by its respective VerbClasses. This approach is similar to lexical generalisation using WordNet, but this time only the verbs are used.

For example, for the verbs “flee” and “escape” the VerbClass is “escape-51.1”, for the verb “arrive” the VerbClass is “escape-51.1-2-1”, which means these verbs have related syntactic frames and are likely to be associated.

Named entity recognition This technique identifies and extracts named entities from each sentence. Unlike other function and content words, named-entities are less likely to be replaced by other words in a plagiarism case. Hence, analysing the number of matching named entities will give a good indication of the topic of texts, and also the similarities between texts.

For example, for the sentences “Albert Einstein is considered to be one of the

[17] http://verbs.colorado.edu/~mpalmer/projects/verbnet.html

most intelligent people that ever lived" and "Numerous awards were given to Albert Einstein, a gifted scientist with great intellectual achievements", the entities "Albert" and "Einstein" are extracted as one entity "Albert Einstein" (person), indicating that both sentences are describing the same person.

The techniques described in this section are commonly applied with other prerequisite techniques. For example, in order to perform **predicate generalisation**, it is first necessary to extract the verbs in the document, then **lemmatise** to generalise the words to their base forms, and finally look up the words in VerbNet for the VerbClass. Another example is the use of **tokenisation**, **lowercasing** and **punctuation removal** in n-gram matching metrics (see Section 4.3).

4.3 Similarity Metrics

In this framework, text pre-processing and shallow NLP techniques are applied before the filtering stage. Deep NLP techniques are applied when the texts have been filtered and further investigation is needed for deeper analysis on candidate texts. This section describes the similarity metrics that are applied after the corpus has been processed. Different similarity metrics are computed depending on the type and level of processing performed. The application of similarity metrics is essential to feature generation, as each feature consists of similarity scores generated by comparing processed text pairs, and the level of similarity for each suspicious-source text pair is determined by the similarity score.

4.3.1 N-gram string matching

The calculation of overlapping n-grams, usually either 3-grams or 5-grams, is a common approach to measuring similarity between texts. 3-grams is normally applied to shorter texts (from several paragraphs to a few pages), and 5-grams is usually used in longer texts (more than a few pages). An n-gram represents n number of consecutive words. Similarity scores can be computed by counting the matching n-grams between the suspicious and source documents. For example, overlapping 3-grams can be exemplified as follows:

Source (Text B): when this man returned he brought me a letter from your father in which he said he was going to try and make his escape and that he would never again set foot in russia

Suspicious (Text A): when this man returned he conveyed me a note from your dad in which he said he was going to trial and make his get away and that he would not ever afresh set base in Russia

Source 3-grams (32 n-grams): [when this man] [this man returned] [man returned he] [returned he brought] [he brought me] [brought me a] [me a letter] [a letter from] [letter from your] [from your father] [your father in] [father in which] [in which he] [which he said] [he said was] [said was going] [was going to] [going to try] [to try and] [try and make] [and make his] [make his escape] [his escape and] [escape and that] [and that he] [that he would] [he would never] [would never again] [never again set] [again set foot] [set foot in] [foot in russia]

Suspicious 3-grams (34 n-grams): [when this man] [this man returned] [man returned he] [returned he conveyed] [he conveyed me] [conveyed me a] [me a note] [a note from] [note from your] [from your dad] [your dad in] [dad in which] [in which he] [which he said] [he said was] [said was going] [was going to] [going to trial] [to trial and] [trial and make] [and make his] [make his get] [his get away] [get away and] [away and that] [and that he] [that he would] [he would not] [would not ever] [not ever afresh] [ever afresh set] [afresh set base] [set base in] [base in russia]

Likewise, overlapping 5-grams will have five tokens instead of three tokens in each n-gram. For example:

Source 5-grams: [when this man returned he] [this man returned he brought] [man returned he brought me] [returned he brought me a letter] ...

Suspicious 5-grams: [when this man returned he] [this man returned he conveyed] [man returned he conveyed me] [returned he conveyed me a] ...

N-grams on their own do not provide an indication as to the level of similarity between two texts. Hence, similarity metrics are needed to calculate the similarity scores between the texts. A similarity metric essentially counts the number of overlapping n-grams between texts, and the count is normalised according to the settings of the experiment.

In the string-matching plagiarism detection system Ferret (Lane et al., 2006), the comparison of n-grams is performed using the Jaccard coefficient:

$$J_3(A, B) = \frac{|S(A, n) \cap S(B, n)|}{|S(A, n) \cup S(B, n)|} \tag{4.1}$$

$S(A, n)$ and $S(B, n)$ represent the sets of n-grams in the suspicious text A and the source text B respectively. In the case of Ferret, $n = 3$. Their intersection (nominator) represents the set of matching n-grams in the suspicious-source text pair, while their union (denominator) represents the set of all distinct n-grams in the suspicious-source text pair.

Using the example described above, the nominator would be 11, using the 3-grams that occur in both sets:

Source 3-grams (32 n-grams): **[when this man]** **[this man returned]** **[man returned he]** [returned he brought] [he brought me] [brought me a] [me a letter] [a letter from] [letter from your] [from your father] [your father in] [father in which] **[in which he]** **[which he said]** **[he said was]** **[said was going]** **[was going to]** [going to try] [to try and] [try and make] [and make his] [make his escape] [his escape and] [escape and that] **[and that he]** **[that he would]** [he would never] [would never again] [never again set] [again set foot] [set foot in] [foot in russia]

Plagiarised 3-grams (34 n-grams):**[when this man]** **[this man returned]** **[man returned he]** [returned he conveyed] [he conveyed me] [conveyed me a] [me a note] [a note from] [note from your] [from your dad] [your dad in] [dad in which] **[in which he]** **[which he said]** **[he said was]** **[said was going]** **[was going to]** [going to trial] [to trial and] [trial and make] **[and**

make his] [make his get] [his get away] [get away and] [away and that] **[and that he]** **[that he would]** [he would not] [would not ever] [not ever afresh] [ever afresh set] [afresh set base] [set base in] [base in russia]

The denominator would be 44, using all the distinct 3-grams from both sets. Hence, the Jaccard coefficient for this example would be $11/44 = 0.25$.

Clough and Stevenson (2010) describe a slightly different similarity metric, the containment measure:

$$c_3(A, B) = \frac{|S(A, n) \cap S(B, n)|}{|S(A, n)|} \tag{4.2}$$

$S(A, n)$ and $S(B, n)$ represent the sets of n-grams in the suspicious text A and the source text B respectively. Similar to the Jaccard coefficient, the containment measure calculates the intersecting n-grams but normalises them only with respect to the n-grams in the suspicious text. This is particularly useful in cases where the suspicious text is shorter than the source text. Using the previous example, the similarity score generated by the containment measure would be $11/34 = 0.32$.

The overlap coefficient is another variant of the Jaccard coefficient, which is also described in Clough and Stevenson (2010):

$$Sim_{Overlap}(A, B) = \frac{|S(A, n) \cap S(B, n)|}{min(|S(A, n)|, |S(B, n)|)} \tag{4.3}$$

Let $S(A, n)$ and $S(B, n)$ be the unique n-grams contained in the suspicious text A and the source text B respectively. The intersection of both sets is divided by the smaller set of $S(A, n)$ or $S(B, n)$. This is useful in cases where the size of suspicious and source text varies. Using the previous example, the similarity score generated by the overlap coefficient would be $11/32 = 0.34$.

In the next chapter, the use of overlapping 3-gram string matching is described in the small-scale experiment using short texts. The use of overlapping n-grams is also a common practice in the PAN competition, where the use of hashed 5-grams has been one of the techniques that contributed to the best approaches (Kasprzak and Brandejs, 2010; Zou et al., 2010). Therefore, in the experiment described in Chapter 6, 5-grams of words are used in detecting similarity for longer texts. The similarity metric overlap coefficient (Formula 4.3) is extensively used.

The framework will exploit the commonly used Jaccard coefficient(Formula 4.1), overlap coefficient(Formula 4.3) and containment measure(Formula 4.2) in generating linguistic features to enhance the variety in the representation.

4.3.2 Language model

Statistical language modelling aims to build a model that can estimate the distribution of natural language texts, considering short sequences of up to n words. An example of toolkit that allows to build such models is SRILM[18] (Stolcke, 2002). In the context of plagiarism detection, based on a model built from one or more source texts, language modelling tools are helpful by estimating the likelihood of a new sequence of words in a suspicious text according to such a model. In other words, a language model can be seen as a measure of how similar the two texts are by comparing their n-gram distributions. We use a standard n-gram language

[18] http://www.speech.sri.com/projects/srilm/

model that computes the probability of a given word based on the sequence of previous $n-1$ words, as opposed to all previous words in a document:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1}) \tag{4.4}$$

This language model will compute $P(w|w-1, w-2..., w-n)$, where w_i are the sequences of words in the suspicious text from 1 to n. The probabilities are estimated using frequencies in the source texts.

Another method is to compute a variant of the language model probability, the perplexity, which normalises the probability scores from language models according to the number of words in the suspicious text.

$$\frac{1}{m} log_2 P(w_1^m) \tag{4.5}$$

Finally, the out-of-vocabulary rate is computed by counting the number of words in the suspicious text that have not been seen in the source texts.

4.3.3 Longest common subsequence

Another string matching metric proposed for the framework is the Longest Common Subsequence (LCS) algorithm (Wise, 1993), which finds the longest sequence of word matches in both suspicious and source texts.

$$Sim_{LCS}(A, B) = log_2 \left(1 + \frac{|LCS(A, B)|}{|B|}\right) \tag{4.6}$$

where A and B are the suspicious and source texts respectively. The set $LCS(A, B)$ is the length of the longest chunk of text in A and B.

The LCS algorithm can be implemented by comparing text pairs using sentence level pair-wise comparisons, among all sentences in both texts, and returning the longest matching sequence between the sentence pairs in a given text. The algorithm returns the following:

- Number of matching words in the text pair;
- Average length of matching words in the text pair;
- Number of matching words in each sentence pair;
- Average length of matching words per sentence pair;
- Total word and sentence count for each text.

The LCS algorithm is known to be complex and very resource-dependent. In this study the Python implementation[19] of LCS is used. The aim of this study is not to find the most efficient algorithm, but rather to explore algorithms which may aid plagiairsm detection.

The text pre-processing, and shallow and deep NLP techniques generates similarity scores for each text pair, and these scores are then represented as features for machine learning in the classification stage. The following similarity metrics are used in the feature generation stage where processed text pairs are compared for a similarity score. The similarity metrics listed below are used in correspondence with the NLP processing techniques described in the previous section.

[19] http://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Longest_common_subsequence

4.3.4 Lexical generalisation

To calculate the similarity between WordNet synsets in text pairs, all synsets are selected for each word in the texts as the comparison key. The synsets from the suspicious text are then compared with the synsets of the the source text to compute the level of similarity, normalised by the total number of synsets from both suspicious and source texts, using the Jaccard coefficient (Formula 4.1) where $n = 1$, having each synset represented as a 1-gram. To count a match between suspicious and source texts, at least one of the synsets corresponding to the possible meaning of the word has to match.

$$Sim_{WordNet}(A, B) = \frac{|S(A, n) \cap S(B, n)|}{|S(A, n) \cup S(B, n)|} \tag{4.7}$$

Where $S(A, n)$ and $S(B, n)$ are the unique synsets representing the suspicious and source texts respectively. The intersection of both sets is divided by the union of $S(A, n)$ and $S(B, n)$.

4.3.5 Syntactic constituent extraction

To calculate the similarity between syntactic constituents of text pairs, the number of intersecting syntactic constituents in the suspicious-source text pair is normalised by the number of syntactic constituents in the suspicious text, using the containment measure (Formula 4.2) where $n = 1$, having each syntactic constituent represented as a 1-gram.

$$Sim_{Constituents}(A, B) = \frac{|S(A, n) \cap S(B, n)|}{|S(A, n)|} \tag{4.8}$$

Let $S(A,n)$ and $S(B,n)$ be the unique syntactic constituents, for example `subj-[inheritance-n-sn]`, contained in the suspicious and source texts respectively. The intersection of both sets is divided by the number of syntactic constituents in suspicious text $S(A,n)$.

4.3.6 Dependency relation extraction

For the calculation of similarity between dependency relations in text pairs, the dependency relations in the suspicious text are compared against those in the source text to check for dependency overlaps between the two texts. The total of matching pairs is computed using the overlap coefficient where n=1, having each dependency relation represented as a 1-gram:

$$Sim_{Dependency}(A,B) = \frac{|S(A,n) \cap S(B,n)|}{min(|S(A,n)|,|S(B,n)|)} \tag{4.9}$$

Let $S(A,n)$ and $S(B,n)$ be the unique dependency relations, for example `det(concept-3, a-1)`, contained in the suspicious and source texts respectively. The number of overlapping relations is normalised by the smaller set of $S(A,n)$ or $S(B,n)$.

4.3.7 Predicate extraction

To compute the number of matching predicates in text pairs, verbs are extracted from both suspicious and source texts, without using the VerbNet generalisation process. The number of intersecting verbs in a text pair is normalised using the overlap coefficient:

$$Sim_{Predicates}(A,B) = \frac{|S(A,n) \cap S(B,n)|}{min(|S(A,n)|,|S(B,n)|)} \tag{4.10}$$

Where $S(A, n)$ and $S(B, n)$ are the unique verbs contained in the suspicious and source texts respectively. The intersection of both sets is divided by the smaller set of $S(A, n)$ or $S(B, n)$.

4.3.8 Predicate generalisation

To compute the number of matching predicate classes in text pairs, verbs extracted from texts are queried for their respective VerbClass using VerbNet. The number of intersecting VerbClasses in the suspicious-source text pair is then normalised using the overlap coefficient:

$$Sim_{VerbClass}(A, B) = \frac{|S(A, n) \cap S(B, n)|}{min(|S(A, n)|, |S(B, n)|)} \tag{4.11}$$

Where $S(A, n)$ and $S(B, n)$ are the unique VerbClasses, for example the verbs “flee” and “escape” belongs to the VerbClass `escape-51.1`, contained in the suspicious and source texts respectively. The intersection of both sets is divided by the smaller set of $S(A, n)$ or $S(B, n)$.

4.3.9 Named entity recognition

For the similarity calculation based on named entities, all named entities from both suspicious and source texts are extracted and then the number of intersecting named entities in the text pair is normalised using the overlap coefficient:

$$Sim_{NameEntity}(A, B) = \frac{|S(A, n) \cap S(B, n)|}{min(|S(A, n)|, |S(B, n)|)} \tag{4.12}$$

Where $S(A, n)$ and $S(B, n)$ are the unique named-entities in the suspicious and source texts respectively. The intersection of both sets is divided by the smaller set of $S(A, n)$ or $S(B, n)$.

4.3.10 Word alignment

Word alignment based on exact word, stemmed word, synonym, and paraphrase is performed at a passage level using the tool METEOR[20], an automatic machine translation evaluation metric (Denkowski and Lavie, 2011).

This metric offers a way to align words and phrases even if they have been paraphrased. For the passage-level experiment described in Section 6.3, this metric gives weighted scores for text pairs depending on the level of resemblance of the texts. For instance, using the “ranking” feature in METEOR, the default weights are assigned corresponding to the four “modules” as follows: 1) exact words with a weight of 1.0; 2) stemmed words with a weight of 0.6; 3) synonyms with a weight of 0.8; and 4) paraphrases with a weight of 0.6. In other words, if the words or phrases in the sentence pair match exactly, it will receive a higher score. If the sentence pair has synonyms in common, the score will be reduced slightly. Texts are normalised with tokenisation and lowercasing within the METEOR framework.

For the experiment on plagiarism detection at the passage level, each suspicious text segment (each case is treated as one passage regardless of how many sentences it contains) is compared against all source text passages. Each text pair will be assigned four scores associated with the four different modules. For example:

Suspicious text A: The majority of sea water is simply pure water, with other substances mixed in. The most well known of these other substances is salt. Salt is made up of molecules, which are made up of sodium and chlorine

[20] http://www.cs.cmu.edu/~alavie/METEOR/

atoms. This is the reason for sea water being 1.1 percent sodium and 2.1 percent chlorine. In addition to salt, there are other atoms present in sea water. Obviously, sea water contains all of the substances which the waters of the earth dissolve and carry down. But, these substances are present in insignificant amounts.

Source text B: Most of sea water, therefore, is just water, that is, pure water. But it contains some other substances as well and the best known of these is salt. Salt is a substance the molecules of which contain atoms of sodium and of chlorine. That is why sea water is about 1.1 percent sodium and about 2.1 percent chlorine. There are some other kinds of atoms in sea water, as you would expect, for it gets all the substances which the waters of the earth dissolve and carry down to it but they are unimportant in amounts.

	Suspicious text		**Source text**		
Module	Content	Function	Content	Function	Total match
Exact words	34	39	34	39	73
Stemmed words	0	0	0	0	0
Synonyms	2	0	2	0	2
Paraphrases	4	4	3	5	8

Table 4.1: Module statistics between the example texts

Precision, recall, f-score and a fragmentation penalty are generated for each text pair, and then the scores are normalised into a final score. For the example text pairs, a final score of 0.35 is found (METEOR varies from 0 to 1). The module statistics for the example texts are shown in Table 4.1, and some examples of the aligned words are showed in Table 4.2.

	Suspicious text	**Source text**
Synonym	Simply (token 7)	Just (token 9)
Paraphrase	In addition (tokens 61 & 62)	As well (tokens 24 & 25)
Paraphrase	. This (tokens 44 & 45)	. That (tokens 50 & 51)
Paraphrase	Obviously (token 75)	Best known (tokens 28 & 29)

Table 4.2: Examples of word alignment

In subsequent experiments, for each suspicious case the top ten source cases with the highest final scores are extracted to be used as features in the machine learning classification.

4.4 Machine Learning Classifiers

To give each candidate text pair a classification, the proposed framework uses the similarity scores generated from the similarity metrics as features (also called *attributes*). The machine learning toolkit used is WEKA[21] version 3.6.5, which provides many different learning algorithms.

All features are normalised by scaling each data variable into a range of 0 to 1, using the WEKA unsupervised attribute normalisation filter[22]:

$$x_{ij}norm = \frac{x_{ij} - x_j^{min}}{x_j^{max} - x_j^{min}} \tag{4.13}$$

where x_{ij} is the feature to be normalised, x_j^{min} is the minimum value and x_j^{max} is the maximum value.

In order to select the best combination of features, the InfoGain attribute

[21] http://www.cs.waikato.ac.nz/ml/weka/

[22] http://weka.sourceforge.net/doc.dev/weka/filters/unsupervised/attribute/Normalize.html

evaluator is used to rank them according to their performance. It evaluates the value of an attribute by measuring the information gain with respect to the class (target attribute, e.g., plagiarism vs clean). The pseudocode for InfoGain is as follows[23]:

```
infoGain(examples, attribute, entropyOfSet)
gain = entropyOfSet
for value in attributeValues(examples, attribute):
sub = subset(examples, attribute, value)
gain -= (number in sub)/(total number of examples) * entropy(sub)
return gain
```

(4.14)

where the decrease in entropy from the original dataset is measured based on the use of a given feature.

Once a set of features is selected, a machine learning model is then built to predict a class for each text pair (an instance), such as a binary classification as “plagiarised” or “clean”.

Machine learning allows the classification of text pairs based on a combination of features generated by more than one similarity metric, which enables a more flexible approach and is much more beneficial than classifying a text pair based on only one similarity metric with a predetermined threshold.

One of the algorithms used is the Naïve Bayes classifier (Formula 4.15), which

[23] http://web.cs.swarthmore.edu/~meeden/cs63/f05/id3.html

is based on the Bayes theorem, where features are assumed to be independent and combined through a probabilistic model:

$$classify(f_1, \ldots, f_n) = argmax_c p(C = c)\Pi_{i=1}^{n} p(F_{i=}f_i|C = c) \tag{4.15}$$

C is the text class and $f_1 ... f_n$ are the features representing examples of how the instance is classified. The classifier considers all features and chooses the most probable hypothesis that can maximise the decision outcome. This algorithm has been applied to other statistical tasks with significant success, such as machine translation (Munteanu and Marcu, 2005) and data mining (Mitchell, 1999). It is considered one of the simplest and yet most effective approaches for empirical NLP (Bao et al., 2004).

Another algorithm is the J48 classifier, a Java implementation of the C4.5 algorithm (Formula 4.16). The algorithm is used to generate a decision tree, which iteratively chooses one attribute that most effectively splits the set of instances into subsets that are more likely to be classified into one class or the other. The feature with the highest confidence in each node is chosen to make the decision, and the process recurs in the smaller subsets.

The pseudocode for building a decision tree is as follows (Kotsiantis, 2007):

```
1 Check for training examples
2 For each attribute A
3 Find the normalized information gain from splitting on A
4 Let $a\_best$ be the attribute with the highest normalized information gain
5 Create a decision node that splits on $a\_best$
6 Recur on the subsets obtained by splitting on $a\_best$,
  and add those nodes as children of node
```

(4.16)

In the plagiarism direction identification experiment, the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm is used in the classification task. This propositional rule-based learner performs well on large and noisy data. It begins with parting the training examples into two subsets, and then adds one condition at a time to the current rule for maximising an information gain measure, until it covers no negative examples.

$$Gain(R', R) = s * \left(log_2 \frac{N'_+}{N'} - log_2 \frac{N_+}{N} \right) \tag{4.17}$$

where R is the original rule, R' is the candidate rule after adding a condition, and s is the number of true positives in the rules after the condition is added. The procedure tries every possible value of each feature and chooses the highest condition based on its InfoGain score. N represents the number of instances that are covered by R, N' represents the candidate instances while N'_+ represnts the positive candidate instances, and N'_+ represents the number of true positives in R.

Another algorithm is the Support Vector Machines (SVM) which performs well on datasets with many features. This kernel-based algorithm transforms the input data to a vector space that can handle many features. The two variations of SVM that are used are Structured Prediction Tree Kernel (SVM-tree kernels) (Moschitti et al., 2006) and the ranker SVM-rank (Joachims, 2006).

For SVM-tree kernels, we use SVM-light-TK[24], an extension of SVM-light[25]. The similarity between partial syntactic trees is measured in terms of their substructures, and the Tree Kernel-based algorithm selects the best substructures that describe the class. The Tree Kernels can be tested with the syntactic tree as a single feature, or with additional features added as vectors of the tree.

$$K(T_1, T_2) = \sum_{n_1 \in N_{T1}} \sum_{n_2 \in N_{T2}} \Delta(n_1, n_2) \tag{4.18}$$

where N_{T1} and N_{T2} are the sets of the tree nodes and $\Delta(n_1, n_2)$ is the number of levels in the sub-tree. The algorithm assigns lower weight to larger text fragments.

For the plagiarism direction ranking task, the SVM-rank[26] ranker tool (Joachims, 2006) is used. The tool adopts a linear classification rule that helps to determine the level of similarity between the texts, and ranks the texts accordingly.

$$rsv(q, d_i) = \overrightarrow{w} * \Phi(q, d_i) = \sum \alpha^*_{k,l} \Phi(q_k, d_l) \Phi(q, d_j) \tag{4.19}$$

where the learned retrieval function is represented as a linear combination of the feature vectors, and kernels can be used to extend the ranking algorithm to

[24] http://disi.unitn.it/moschitti/Tree-Kernel.htm
[25] http://svmlight.joachims.org/
[26] http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

non-linear retrieval functions. q represents the query and d the documents, w is the weighted vector that determines the ranking, $\Phi(q, d)$ represents the mapping of features between query and document, and $\sum \alpha^*_{k,l}$ measures the pair-wise differences of the vectors.

4.5 Evaluation Metrics

This section describes the evaluation metrics used to test the classification performance on each text pair as a result of the application of the classification models built via machine learning. The section also describes the evaluation metric for assessing individual feature performance.

4.5.1 Correlation coefficient

Pearson's coefficient is used to evaluate the linear dependence between two variables:

$$r = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{X_i - \bar{X}}{s_X} \right) \left(\frac{Y_i - \bar{Y}}{s_Y} \right) \tag{4.20}$$

Two variables X (suspicious texts) and Y (source texts) with the relative frequency of n values in X and Y, represented by X_n and Y_n.The means of X and Y are represented with $\bar{X}$ and $\bar{Y}$. s_X and s_Y are the standard deviation of X and Y.

The advantage of using a correlation coefficient is that the features do not need to be normalised as the correlation is not dependent between features. The features can be evaluated individually in a straightforward manner.

4.5.2 Precision, recall, F-score and accuracy

The standard metrics of precision, recall, F-score and accuracy over the classification results are used for evaluation. The correctly classified plagiarised texts (True Positives: TP), correctly classified clean texts (True Negatives: TN), clean texts incorrectly classified as plagiarised (False Positives: FP), plagiarised texts incorrectly classified as clean (False Negatives: FN) are used in the standard calculation of precision, recall, F-score, and accuracy as follows:

$$Precision = \frac{TP}{TP + FP} \tag{4.21}$$

Precision calculates the number of texts correctly identified as belonging to a class, normalised by the total number of texts both correctly and incorrectly identified as belonging to that class.

$$Recall = \frac{TP}{TP + FN} \tag{4.22}$$

Recall calculates the number of correctly identified texts as belonging to a class, normalised by the total number of correctly identified texts and those that have not been identified as belonging to that class but should have been.

$$F - Score = 2 * \frac{P * R}{P + R} \tag{4.23}$$

F-score is the harmonic mean of precision and recall.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{4.24}$$

Accuracy gives the proportion of the total number of correctly identified documents over all the sets.

4.5.3 Statistical significance

To assess whether the results obtained reflect a pattern rather than just occur by chance, statistical significance is calculated using a two-tailed z-test. The Z-test is used for data with a normal distribution where examples are independent of each other. In this framework $\alpha = 0.05$, where a confidence level of 95% or above brings a statistically significant result.

$$z = \frac{\overline{x_1} - \overline{x_2} - \Delta}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \tag{4.25}$$

where $\overline{x_1} - \overline{x_2}$ is the observed difference , and Δ is the expected difference between the population means. The observed and expected differences are normalised by the standard error for the difference, where σ_1 and σ_2 are the standard deviations of the two populations, and n_1 and n_2 are the sizes of the two samples.

The statistical significance test is applied in the experiments comparing the proposed framework against other PAN approaches, as described in Section 6.3.

4.6 Summary

This chapter described the general framework for the proposed plagiarism detection approach. The text pre-processing, and shallow and deep NLP techniques were explained. The techniques are inspired by related research and brought together for an empirical analysis. The **first objective**, described in Section 1.3, is fulfilled by incorporating shallow and deep NLP techniques as part of a plagiarism detection framework. The description of the techniques was followed by a list of similarity metrics that measure the similarity between texts and generate features to be used in the machine learning classifiers. The similarity metrics include the long-established string-matching algorithms, along with statistical language models and longest common subsequence. Novel linguistic information-matching features such as the incorporation of syntactic constituent extraction, predicate generalisation and named entity recognition are investigated with a supervised machine learning classification, which are underexplored in the plagiarism detection field. The chapter concluded with a list of the conventional evaluation metrics which are used in this analysis.

Chapter 5

Experiments with a Small-scale Corpus

This chapter describes a pilot experiment performed on a small-scale corpus. The main goal of this experiment is to explore both shallow and deep NLP techniques, and to analyse the effects of individual technique as well as combined techniques. This initial experiment identifies the techniques which contribute best in order to build a foundation for further experimentation. Section 5.1 covers the details of the corpus. Section 5.2 describes the text pre-processing and NLP techniques applied in the experiment. Section 5.3 lists the similarity metrics used to generate the features. Section 5.4 presents the results of individual and combined features, and Section 5.5 gives an evaluation of the best features against baseline features. The chapter concludes with a discussion in Section 5.6.

As described in Chapter 3, existing plagiarism detection approaches rely on superficial string-matching metrics. In our study, both superficial and structural approaches are explored to find the best combination of techniques.

5.1 Corpus

There are very few authentic plagiarism cases available for an empirical research. Current plagiarism detection corpora, described in Section 3.3.1, are limited to the automatic substitution of text from the original source document into the suspicious document, with some artificial obfuscations inserted. In early studies, the corpora used in experiments were not tailored to the purpose of plagiarism detection. Examples are the Microsoft Research Paraphrase Corpus[27] and the Measuring Text Reuse Corpus (METER)(Gaizauskas et al., 2001). These corpora do not accurately reflect the types of plagiarism that are present in a real-case scenario, thus they are not best suited for plagiarism detection experiments. In the tests on plagiarism detection software performed by Weber-Wulff (2010), manually created plagiarised texts were used, but the majority of the samples are not in English and the document collections are far too small for a quantitative evaluation. In order to facilitate the development and evaluation of plagiarism detection systems, Clough and Stevenson (2010) constructed a corpus consisting of various levels of plagiarism in short texts, which is used in the experiment described in this section.

To test the framework proposed in Chapter 4, the small-scale corpus by Clough and Stevenson (2010) was chosen. The corpus consists of short texts written by students, with three levels of rewriting that replicate common characteristics of plagiarism. The corpus contains five source documents and 95 suspicious documents. The suspicious documents are short texts that contain several hundred words and the source documents are excerpts from Wikipedia computer science articles. The suspicious documents include 57 plagiarised and 38 clean (non-plagiarised) cases. Each suspicious document corresponds to one source document only.

[27] http://research.microsoft.com/en-us/downloads/607d14d9-20cd-47e3-85bc-a2f65cd28042/

Document class	**Attribute**	**Statistics**
Source documents	Number of documents Minimum length Maximum length Average length	5 289 words 594 words 441.5 words
Suspicious documents (overall)	Number of documents Minimum length Maximum length Average length	95 43 words 406 words 224.5 words
Suspicious documents (Clean)	Number of documents Minimum length Maximum length Average length	38 43 words 332 words 187.5 words
Suspicious documents (Heavy revision)	Number of documents Minimum length Maximum length Average length	19 107 words 387 words 247 words
Suspicious documents (Light revision)	Number of documents Minimum length Maximum length Average length	19 87 words 384 words 235.5 words
Suspicious documents (Near copy)	Number of documents Minimum length Maximum length Average length	19 119 words 406 words 262.5 words

Table 5.1: Corpus statistics

Table 5.1 lists the details of the corpus. As shown in the table, there are three levels of rewriting amongst the plagiarised documents, and the required length is between 200-300 words:

- Near copy: cases where answers were directly copied from the original article but without instructions on which parts of the article to copy.
- Light Revision: answers were copied from the original article with minor alterations, such as paraphrasing, but sentence structures were not changed.

- Heavy Revision: answers were based on the original article but were rephrased and altered with different words and structures.

The clean cases were written without reference to the original article, and the answers were based on the author's own knowledge and wordings.

In this experiment, a multiclass classification and a binary classification are adopted, which is described in the following section.

5.2 Text Pre-processing and NLP Techniques

The corpus was processed with the five-stage framework described in Section 4.1, which include the pre-processing stage, the similarity comparison stage, the filtering stage, the further processing stage, and the classification stage.

The final stage is to use the similarity scores generated from the similarity comparison stage to give each document pair a binary classification of *Plagiarised* or *Clean*, or a multiclass classification for each document pair in four levels: *Clean*, *Near Copy*, *Heavy Revision*, or *Light Revision*. The classification is either done by setting thresholds, or by using similarity scores as features in a machine learning classifier. Document classifications are evaluated by applying evaluation metrics.

The text pre-processing techniques include:

- Sentence segmentation
- Tokenisation
- Lowercasing
- Stopword removal

- Punctuation removal
- Number replacement

The shallow NLP techniques include:

- Part-of-Speech Tagging
- Stemming
- Lemmatisation
- Chunking

The deep NLP techniques include:

- Lexical generalisation
- Syntactic constituent extraction
- Dependency relation extraction

Some shallow NLP or deep NLP techniques have prerequisite text pre-processing techniques, for example, for lexical generalisation, the text processing techniques of tokenisation, lowercasing, stopword removal and punctuation removal are required before lexical generalisation can be performed.

After applying these techniques, the output texts were further processed using one of the following similarity metrics (which are further described in Section 5.3):

- Jaccard coefficient (Formula 4.1)
- Containment measure (Formula 4.2)
- Language model probability metric (Formula 4.5)
- Longest common subsequence (Formula 4.6)

- Lexical generalisation (Formula 4.7)
- Syntactic constituent extraction (Formula 4.8)
- Dependency relation extraction (Formula 4.9)

A feature consists of similarity scores generated using a combination of processing techniques and one of these metrics for each instance of a suspicious-source document pair. The features therefore are a representation of the outcome of similarity scores that correspond to a specific set of processed documents. 56 features were generated in total. Table 5.2 shows a description of pre-selected features according to their type of processing (such as text processing, shallow NLP, deep NLP) and Pearson's correlation coefficient scores (Formula 4.20 on page 4.20); a list of all features along with their performances can be found in Appendix B Table B.1.

5.3 Similarity Metrics

For the n-gram string-matching techniques, the corresponding similarity metric uses overlapping 3-gram string matching metrics such as the Jaccard coefficient (Formula 4.1 on page 103) for calculating the similarity scores. Related research shows that the use of 3-grams is the balance between efficiency and effectiveness with short case lengths. Hence, n-grams of three words were chosen for this experiment.

The plagiarism detection tool Ferret (Lane et al., 2006) calculates the similarity of document pairs based on overlapping 3-grams of words using the Jaccard

Feature	Techniques	Similarity Metric(Formula)
1	Ferret system 3-grams of words	Jaccard coefficient (Formula 4.1)
8	Tokenisation, Lowercasing, Lemmatisation, Ferret system 3-grams of words	Jaccard coefficient (Formula 4.1)
25	Tokenisation, Lowercasing, 3-grams of words	Language Model Perplexity (Formula 4.5)
36	Tokenisation, Lowercasing	Longest Common Subsequence (Formula 4.6)
40	Sentence segmentation, Parsing	Dependency relation extraction (Formula 4.9)
43	Tokenisation, Lowercasing, Punctuation removal, Stopword removal, Lexical generalisation	Lexical generalisation (Formula 4.7)
46	Tokenisation, Lowercasing, 3-grams of words	Containment measure (Formula 4.2)
49	Tokenisation, Lowercasing, Lemmatisation, 3-grams of words	Containment measure (Formula 4.2)
55	Syntactic constituent extraction	Syntactic constituent extraction (Formula 4.8)
56	Syntactic constituent extraction (Removed singleton constituents)	Syntactic constituent extraction (Formula 4.8)

Table 5.2: Combinations of techniques and similarity metrics of selected features

coefficient (Formula 4.1 on page 103). It performs default text pre-processing on the input documents, including sentence segmentation, tokenisation and lowercasing. These three techniques formed the original baseline (Feature 1) for this study. Feature sets 2-15 were also processed by Ferret.

The containment measure (Formula 4.2 on page 104) was applied after sentence segmentation, tokenisation and lowercasing as an alternative comparative baseline (Feature 46). This measure is suitable when one set of documents is longer than the other. In this corpus the source documents are always longer than the suspicious documents, which make the containment measure an appropriate alternative baseline.

The linguistic information generated from the NLP techniques was then matched using one of the following similarity metrics. For lexical generalisation (Feature 43), the Jaccard coefficient was applied to measure the number of matching synsets between the candidate document pairs (Formula 4.7 on page 108). Text pre-processing techniques were applied before processing the texts using WordNet. These techniques included tokenisation, lowercasing, punctuation removal and stopword removal. Using WordNet, all synsets related to each word were retrieved. The comparison metric applied was the 1-gram Jaccard coefficient where the number of matching synsets between a suspicious document and a source document was normalised by their union.

For syntactic constituent extraction (Features 55 and 56), the containment measure was applied to measure the number of matching constituents between the candidate document pairs (Formula 4.8 on page 108). The syntactic constituents were extracted from each sentence and the constituents from the documents were compared against each other to calculate the number of matching relations. The documents were pre-processed using tokenisation, lowercasing, lemmatisation and stopword removal, with an additional feature generated by removing all singleton constituents.

For dependency relation extraction (Feature 40), the overlap coefficient was applied to measure the number of matching dependency relations between the candidate document pairs (Formula 4.9 on page 109). To extract the dependency relations, the documents were pre-processed with sentence segmentation. Then the document was parsed to generate the dependency relations for each sentence, which represent the syntactic relations. The number of matching dependency relations between the document was normalised by the smaller set of relations of the suspicious or the source document.

Furthermore, 1-grams, 2-grams and 3-grams language models (Formula 4.4 on page 106) were used to generate baseline scores (Feature 1). Language model scores were also computed on chunked data (Feature 11) by using 4-grams and 5-grams probability distributions. The out-of-vocabulary rate was also computed, which represents the number of words in the suspicious document that are not present in the source document.

In addition, the LCS metric (Formula 4.6 on page 106) was applied with the tokenised and lowercased corpus. LCS was applied to each document pair at the sentence level and checked for:

- the overall longest matching sequence in that document pair;
- the sum of the longest matching sequence for all sentences normalised by the total number of sentences in the suspicious document;
- the average length of matching sequences; and
- the total number of matching words in each sentence pair normalised by all sentences from the document pair.

This resulted in several LCS-based features, but none of the features provided satisfactory correlation or accuracy scores, hence they had not been investigated further.

In order to get a glimpse of how discriminative each feature is in relation to the four levels of plagiarism, Pearson's correlation coefficient (Formula 4.20 on page 118) was applied to the scores generated by the similarity metrics, which were then compared to the annotations of the actual case classes. As the correlation scores are more interpretable than the InfoGain scores in the machine learning model, the features with higher correlations are selected for further analysis. The correlations of various pre-selected feature sets are shown in Section 5.4.

Finally, a machine learning classifier was applied to classify each suspicious-source document pair. Features were normalised to the values between 0 and 1, using the WEKA unsupervised attribute normalisation filter (Formula 4.13 on page 113), before training and testing in the Naïve Bayes 10-fold classifier (Formula 4.15 on page 115), with 95 document pairs and a selection of the 56 features, which are described in the following section.

5.4 Results

Tables 5.3, 5.4 and 5.5 show three groups of feature sets which are listed with their correlation performance. A complete list of features and their correlation scores is available in Appendix B Figure B.1. The results are based on the multi-class task of four-levels: *clean*, *heavy revision*, *light revision* and *near-copy*.

A feature selection metric, InfoGain attribute evaluator (Formula 4.14 on page 114), was used to select the best features according to their classification performance. The correlation coefficient (Formula 4.20 on page 118) was applied to the best-performing features to demonstrate the degree of robustness, as shown in

Table 5.3. The performance of the individual and combined feature sets was evaluated comparatively against the baseline features. Other two sets of features were pre-selected for the analysis: “deep processing features” (Table 5.4) and “baseline features” (Table 5.5).

The “deep processing features set” were manually selected to include deep NLP techniques: lexical generalisation, syntactic constituents, and dependency relations. Similarly, the “baseline features” were manually selected to contain only overlapping n-grams metrics, which are commonly used in related work.

Each feature set is used to feed a classifier using the Naïve Bayes algorithm (Formula 4.15 on page 115).

Feature	**Techniques**	**Correlation**
49	Lemmatised 3-gram containment	0.769
55	Syntactic constituents extraction	0.768
40	Dependency relations extraction	0.760
8	Lemmatised 3-gram Jaccard	0.632

Table 5.3: Best features set

Feature	**Techniques**	**Correlation**
43	Lexical generalisation	0.783
55	Syntactic constituents extraction	0.768
40	Dependency relations extraction	0.760
56	Syntactic constituent extraction (Removed singleton constituents)	0.731

Table 5.4: Deep processing features set

The deep processing feature lexical generalisation achieved the highest correlation with the classes of plagiarism in comparison to other features. However,

Feature	**Feature**	**Correlation**
46	Original baseline - Containment measure	0.768
25	Language Model 3-gram Perplexity	0.671
1	Original baseline - Jaccard coefficient	0.632

Table 5.5: Baseline features set

it was not selected by the InfoGain attribute evaluator for the best features set, as their criteria to measure feature performance are very different. Another deep technique, syntactic constituent extraction, matched the performance of the overlapping 3-gram feature. The best feature set included two 3-gram metrics with lemmatisation, and two deep features. In Table 5.6, the Naïve Bayes classifier with a 10-fold cross validation showed very promising performance for both the best features and the deep processing features. A majority class baseline is set, based on the 38 clean cases in a total of 95 cases, the majority class baseline performance of 40% is shown alongside the other features.

Feature Set	**Accuracy**
Best features (Table 5.3)	**71.58%**
Deep processing features (Table 5.4)	**71.58%**
All features (Appendix B.1)	67.37%
Baseline features (Table 5.5)	67.37%
Baseline	40%

Table 5.6: Document classification accuracy of different feature sets

The best features and deep processing features reached the same accuracy, which indicates that the most contributing features are generated by the deep techniques, as shown by their correlation scores. The deep features outperformed the baseline, and the combination of techniques was shown to be more effective than relying on n-gram matching on its own.

More detailed analyses of precision, recall and F-score for the different classes of the multi-class classification problem are shown in Figures 5.1, 5.2 and 5.3, respectively.

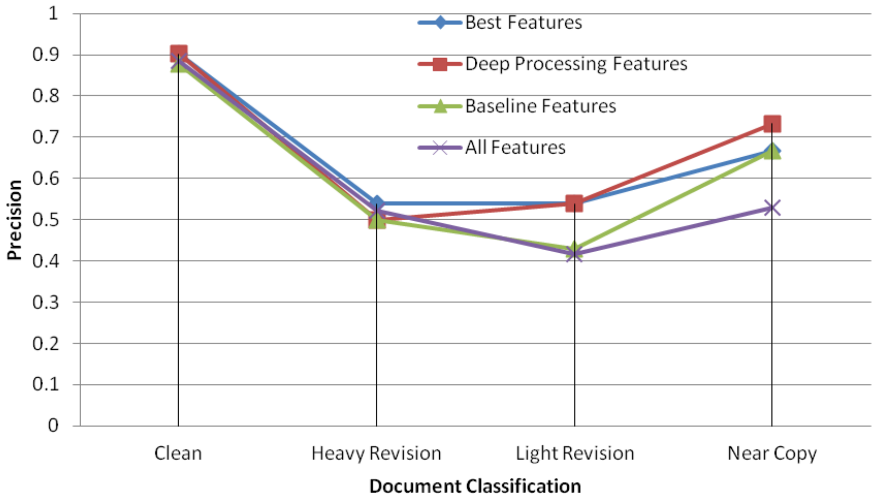

Figure 5.1: Precision for Naïve Bayes document classification with different feature set

Taking a closer look at the comparison between the baseline features and the best features, Table 5.7 lists the precision, recall and F-score for each class with the two features sets.

The results show that classifying cases into four classes is not an easy task. Both feature sets are effective in identifying clean cases, but when it comes to the level of plagiarism the best features outperformed the baseline features in the heavy revision class. For the light revision class, the best features achieved a higher precision but slightly lower recall and F-score. There are no differences in the near

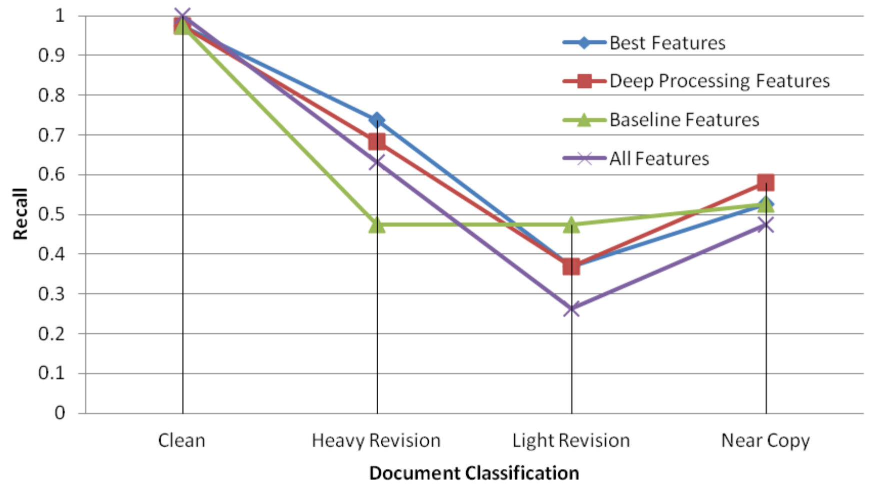

Figure 5.2: Recall for Naïve Bayes document classification with different feature set

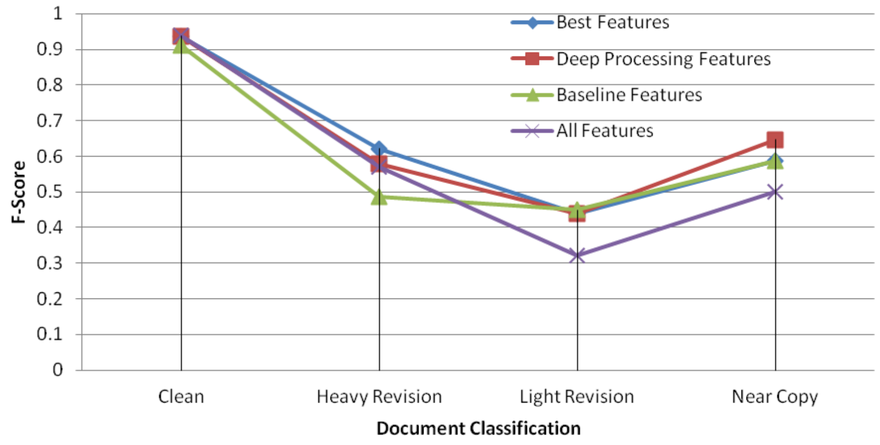

Figure 5.3: F-Score for Naïve Bayes document classification with different feature set

copy class. A trend can be observed that the best features set is more effective in identifying complex plagiarism (heavy revision class), but the results did not show significant differences in other classes.

	Precision		**Recall**		**F-score**	
Class	Best Features	Baseline Features	Best Features	Baseline Features	Best Features	Baseline Features
Clean	**90.2%**	87.8%	97.4%	97.4%	**93.7%**	91.1%
Heavy	**53.8%**	50%	**73.7%**	47.4%	**62.2%**	48.6%
Light	**53.8%**	42.9%	36.8%	**47.4%**	43.8%	**45%**
Copy	66.7%	66.7%	52.6%	52.6%	58.8%	58.8%

Table 5.7: Comparison of precision, recall and F-score of best features against baseline features in four-class classification

In comparison, the binary classification task is much more promising. Table 5.8 shows the results obtained based on a binary classification of *Clean* and *Plagiarised*.

	Precision		**Recall**		**F-score**	
Class	Best Features	Baseline Features	Best Features	Baseline Features	Best Features	Baseline Features
Clean	**92.7%**	90%	**100%**	94.7%	**96.2%**	92.3%
Plag	**100%**	96.4%	**94.7%**	93.0%	**97.3%**	94.6%

Table 5.8: Comparison of precision, recall and F-score of best features against baseline features in binary classification

The binary classification using the best features was very promising, achieved an overall accuracy of 96.8%, and 100% recall in the *Clean* class and 100% precision in the *Plagiarism* class. The baseline features have also achieved a promising overall accuracy of 93.7%, but the size of the corpus is rather small to allow further analysis.

5.5 Discussion

The trade-off between speed and reliability is noticeable in this initial experiment. For instance, the dependency relations feature (Feature 40) required a longer processing time than any shallow approach, as it requires each document to be preprocessed, parsed and then post-processed to extract the relations. Nevertheless, dependency relations turned out to be one of the most promising features, with a high correlation coefficient, and also one of the most contributing features in the machine learning classifier. Shallower techniques did not perform as well.

The use of deep NLP techniques showed improvement in identifying the heavy revision class, but it is possible that the results are only significant as the corpus is small and the plagiarised cases are relatively easy to identify. Therefore, further experiments are needed to test the techniques on a large scale corpus in order to investigate the practicability of using deep NLP techniques in plagiarism detection. These experiments are described in Chapter 6.

The result of the binary classification was very promising. When classifying the documents into two classes of *Plagiarised* and *Clean*, all clean documents were correctly classified, and only three out of 57 plagiarised documents were incorrectly classified as clean.

On the other hand, distinguishing amongst the three different levels of plagiarism turned out to be a much more complex task. However, since the identification of the plagiarism level is not the main focus of this experiment, the results are already very promising and provide a good indication for the direction of further experiments.

For a closer inspection, excerpts of plagiarism cases are given below. The cases are a computer science article explaining the principle of VSM.

Source document B "Vector space model (or term vector model) is an algebraic model for representing text documents (and any objects, in general) as vectors of identifiers, such as, for example, index terms. It is used in information filtering, information retrieval, indexing and relevancy rankings. Its first use was in the SMART Information Retrieval System. A document is represented as a vector. Each dimension corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. Several different ways of computing these values, also known as (term) weights, have been developed. One of the best known schemes is tf-idf weighting (see the example below)..."

Non-plagiarised document (Clean) A1 "Within Information Retrieval each document in a set can be represented as a point in high-dimensional vector space, this representation is called the vector space model. Information Retrieval queries are also represented as vectors in the same vector space; these are then used in conjunction with the document vectors to find relevant documents..."

Plagiarised document with heavy revision (Heavy Revision) A2 "There are a large number of models used in solving the problem of Information Retrieval and they are all based on one of three mathematical bases: set theory, algebra and probabilistic. The vector space model is one of these methods, and it is an algebraic model. In the vector space model a document is represented as a vector. Within this vector, each dimension corresponds to a separate term (where a term is typically a single word, keyword or phrase.)..."

Plagiarised document with light revision (Light Revision) A3 "The vector space model (also called, term vector model) is an algebraic model used to represent text documents, as well as any objects in general, as vectors of identifiers. It is used in information retrieval and was first used in the SMART Information Retrieval System. A document is represented as a vector and each dimension corresponds to a separate term. If a term appears in the document then its value in the vector is non-zero. Many different ways of calculating these values, also known as (term) weights, have been developed. One of the best known methods is called tf-idf weighting..."

Plagiarised document with cut-and-paste text (Near Copy) A4 "Vector space model is an algebraic model for representing text documents (and in general, any objects) as vectors of identifiers, such as, for example, index terms. Its first use was in the SMART Information Retrieval System. It is used in information filtering, information retrieval, indexing and relevancy rankings. A document is represented as a vector, and each dimension corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. Several different ways of computing these values, also known as (term) weights, have been developed..."

An analysis of how the various similarity metrics performed on these examples of documents is presented in Table 5.9.

In the experiments described so far, similarity scores are used as features in conjunction with the machine learning classifier. For the purpose of this analy-

	3-grams Lemmatised Jaccard (Formula 4.1)	**3-grams Containment (Formula 4.2)**	**Lexical generalisation (Formula 4.7)**	**Syntactic constituent extraction (Formula 4.8)**	**Dependency relation extraction (Formula 4.9)**
A1 Clean	0.01	0.02	0.37	0.02	0.02
A2 Heavy	0.08	0.14	0.60	0.24	0.16
A3 Light	0.43	0.61	0.89	0.43	0.63
A4 Copy	0.61	0.82	0.90	0.64	0.72

Table 5.9: Comparisons between various features on the four examples

sis, the raw similarity scores are listed. The scores are based on the comparison between the source document B and suspicious documents A1, A2, A3 and A4. From the analysis, it is clear that even simple string-matching techniques in the first and second column are able to achieve good results on this corpus. The overlapping 3-gram metrics with text pre-processing and shallow NLP techniques are able to distinguish between clean and plagiarised documents with ease. On the other hand, deeper techniques may be more useful in helping to distinguish between different levels of plagiarism, as it can be seen by the often larger differences in the scores for different levels of plagiarism. Without a machine learning classifier, thresholds would need to be set in order to determine the level of plagiarism for each example. It would be a complex task to identify the suitable threshold for each level and for each feature. In that sense, the application of a machine learning classifier was a rational solution to avoid ad hoc decisions.

The experiment described in this chapter has attracted some interests. Bär et al. (2012) investigated text reuse detection using several corpora, including the Clough & Stevenson corpus. Their approach investigates content, structural and stylistic features. The best combinations are content features which include Longest Common Substring (LCS) and 2-grams of words, and also structural features which include lemmatised word order and distance measures, and n-grams of stopwords. Their results on the binary classification achieved an accuracy of 96.8%, which matches our results, and their results on four-class classification achieved an accuracy of 84.2% which outperforms our results by 12.6%.

Sánchez-Vega et al. (2013) also investigated text reuse using the Clough & Stevenson corpus. Their approach assigns a weight to the words within a document to analyse the relation between them. This approach can capture changes in structure by characterising the document with features such as the degree of rewriting, relevance and fragmentation. These features are determined by the number of words copied from source and the length of the copied texts. Their results on a four-class classification achieved an accuracy of 75.9% which outperforms our results by 4.3%.

The above studies showed that although string-matching techniques are superficial, they could be improved by analysing the structure of the texts using features such as word order, word relevance and word distance. Currently, our framework lacks structural analysis of this kind; this would be an interesting direction for future work.

5.6 Summary

Besides the inherent complexity of the task, the relatively low accuracy in distinguishing different levels of plagiarism may be due to some characteristics of the corpus. Particularly, during the corpus creation stage, not all participants seemed to have followed the instructions given to them to rewrite the short texts. For example, some cases annotated as *Near copy* plagiarism actually contained some revised passages, and should therefore have been annotated as *Light revision.*

The main contribution of this experiment is the application of shallow and deep NLP techniques and the incorporation of machine learning classification in plagiarism detection. The framework has been tested with various combinations of shallow and deep techniques. The results showed that some of the most promising techniques are deep linguistic analyses based on lexical generalisation, dependency relations extraction and syntactic constituent extraction. For future work, it may be possible to apply a parser for other languages to achieve similar performance in cross-lingual plagiarism detection tasks.

This chapter described a small-scale experiment which applied text preprocessing, and shallow and deep NLP techniques in plagiarism detection. The techniques are investigated individually and as combined feature sets, using similarity metrics and machine learning classifiers to evaluate their effectiveness. This experiment achieved the **second objective** of the thesis, mentioned in Section 1.3, in which the proposed framework of incorporating string-matching and NLP techniques is evaluated using an empirical approach. This initial experiment identified notable NLP techniques that are promising in a small-scale scenario. The experiment was performed at the document level, although the document length is short (maximum 300 words) and therefore the texts could be regarded as passages. The next research question is to find out whether the framework is applicable in a large-scale scenario with longer case lengths, which is essential for the development of a realistic plagiarism detection approach. Taking the document length into consideration, the next chapter describes another experiment performed at the document level and at the passage level on a larger corpus.

Chapter 6

Experiments with a Large-scale Corpus

This chapter describes three experiments performed on different subsets of the PAN plagiarism corpus 2010 (PAN-PC-10) (Potthast et al., 2010c). Section 6.1 presents an initial experiment on a subset of 1,000 suspicious documents. Section 6.2 details a further experiment using all manually simulated documents. Section 6.3 describes an experiment performed at the passage level on manually simulated cases.

6.1 Document-level Initial Experiment

This section describes the initial experiment performed on a small subset of PAN-PC-10 corpus. The detection was performed at the document level on both artificially generated and manually simulated cases. The experiment follows the five-stage detection framework proposed in Section 4.1.

6.1.1 Corpus

The PAN-PC-10 corpus is by nature very different from the Clough & Stevenson corpus (Clough and Stevenson, 2010) used in the previous chapter. It features a much larger document collection, longer document length and contains plagiarism cases artificially created by automatic extraction of texts. In some cases, automatic obfuscation techniques were involved to generate rewritten texts, and some of the original texts are not available in the document collection, meaning that online detection that searches within the document collection as well as the web is required.

The corpus has a total of 11,147 source and 15,925 suspicious documents. 70% of the plagiarism cases involve external detection. The remaining 30% are intrinsic detection cases and online detection cases which are out-of-scope for this study. 40% of the plagiarism cases are verbatim copies from multiple sources (no obfuscation). The other 40% of cases contain artificially inserted passages with two levels of automatic obfuscation, low or high, achieved by applying obfuscation techniques as described in Potthast et al. (2010c):

Original text "The quick brown fox jumps over the lazy dog."

Manual rewrite "Over the dog which is lazy jumps quickly the fox which is brown."

Random text operations "over jumps quick brown fox The lazy. the"

Semantic word variation "The quick brown vixen leaps over the lazy puppy."

POS-preserving word shuffling "The brown lazy fox jumps over the quick dog."

A plagiarised text with "low" obfuscation indicates that only one or two techniques was applied, whereas a text with "high" obfuscation was processed with a combination of techniques.

A small number of cases (6%) are simulated plagiarism cases where texts were manually rewritten with different wordings using Amazon Mechanical Turk. The remaining cases (14%) are translated plagiarism texts from Spanish or German to English.

The source collection contains English, Spanish and German documents. In this experiment, the scope is set for monolingual detection, thus it was essential to identify and filter out non-English documents from further processing stages. An automatic language identifier, TextCat[28], was used for this purpose and there were 10,416 source documents identified as English. The 1,001 non-English documents were excluded from the corpus.

As the experiment was a preliminary study, the first 1,000 suspicious documents were selected along with the 10,416 source documents for further processing. To investigate external plagiarism, all intrinsic and translated plagiarism cases were excluded from the dataset. 186 cases were removed from the first subset of 1,000 suspicious documents. The experiments presented in the subsequent sections were therefore based on 814 suspicious documents and 10,416 source documents, which gives a total of 8,478,624 possible pair-wise comparison cases.

Document class	**Attribute**	**Statistics**
Source document	Number of documents	10,416
Suspicious	Number of documents	814
	Minimum case length	50 words
	Maximum case length	5,000 words

Table 6.1: Corpus statistics

Table 6.1 shows the corpus statistics. The plagiarism segments in a suspicious

[28] TextCat Language Guesser http://odur.let.rug.nl/~vannoord/TextCat/

document can come from one to more than 50 sources. A third of the plagiarised cases are short, between 50 and 150 words; another third of the cases are between 300 and 500 words, and the remaining third of the cases contain long plagiarised texts which are between 3,000 and 5,000 words. The length of documents ranges from 1 page to 1,000 pages, and most of the suspicious documents contain less than 20% plagiarised text.

The evaluation method used in this experiment follows a binary classification: *plagiarised* or *clean*. In the annotation provided by the PAN-PC-10 corpus, plagiarised cases were annotated at the passage level. However, at this stage, cases were treated at the document level rather than passage level, where a pair of documents will be considered as plagiarised whenever at least one passage within the documents matched.

6.1.2 Text Pre-processing and NLP Techniques

Following the five-stage approach described in Section 4.1, the selected corpus was processed using the techniques as shown in Table 6.2 for generalising texts. The most promising techniques which showed the best performance from the small-scale experiment (Chapter 5) were further investigated in this experiment. In this section, the text pre-processing techniques used included tokenisation, lowercasing, sentence segmentation and punctuation removal. Shallow NLP techniques included stemming. Deep NLP techniques included lexical generalisation and dependency relation extraction. These techniques were shown to be of particular interest in the previous chapter.

In order to detect semantically-related words in texts, lexical generalisation was performed on the word level, where function words (stopwords) were removed and all remaining (content) words were replaced by their synsets from WordNet. As all senses of each word were selected, Word Sense Disambiguation was not needed. For example, the overlapping synsets of the words `jump` and `leap` are counted as a match. Similarity is determined by the number of overlapping synsets between two texts.

To investigate the structural changes in texts, dependency relations are extracted, where the corpus was first pre-processed with sentence segmentation to determine sentence boundaries in documents. Then a parser was applied to generate dependency relations of each sentence. For example, the dependency relation `nsubj(jumps, fox)` in the previous section are counted as a match. Similarity is determined by the number of matching dependency relations between two texts.

The comparative baseline was processed with overlapping n-grams metrics with two variations, where texts were split into 5-grams of words, within sentence boundaries or across sentence boundaries. This is to investigate whether sentence segmentation would affect detection performance in longer texts.

One of the comparative baselines was Feature 1, using 1-grams on the original corpus without any pre-processing. Another baseline comparative feature was Feature 2 which used 5-grams on a tokenised and lowercased corpus.

Feature	Techniques	Similarity Metric(Formula)
1	1-gram of words	Overlap coefficient (Formula 4.3, p.104)
2	Tokenisation, Lowercasing , 5-gram of words	Overlap coefficient (Formula 4.3, p.104)
3	Sentence segmentation, Tokenisation, Lowercasing, Punctuation removal, Stemming, 5-gram of words (within sentence boundaries)	Overlap coefficient (Formula 4.3, p.104)
4	Tokenisation, Lowercasing, Punctuation removal, Stemming, 5-gram of words (across sentence boundaries)	Overlap coefficient (Formula 4.3, p.104)
5	Tokenisation, Lowercasing, Punctuation removal, Stopword removal, Lexical generalisation	Lexical generalisation (Formula 4.7, p.108)
6	Sentence segmentation, Dependency relation extraction	Dependency relation extraction (Formula 4.9, p.109)

Table 6.2: Pre-processing and NLP techniques applied in each feature

6.1.3 Similarity Metrics

In the second stage of the framework, similarity metrics are applied to processed texts to compute similarity scores between each suspicious-source document pair.

The use of overlapping n-grams was a common practice in the PAN competition, and use of hashed overlapping 5-grams was one of the techniques contributing to the best approaches (Kasprzak and Brandejs, 2010; Zou et al., 2010). Therefore, n-grams of five words were chosen for this experiment. N-gram-based features were applied with the overlap coefficient (Formula 4.3 on page 104) to count the number of 5-grams the document pairs have in common. 5-grams of words were applied in this experiment to Features 2, 3 and 4 as the document length is long.

For the experiment in lexical generalisation (Feature 5), the synsets from suspicious documents were compared against those in source documents. To count a match between suspicious and source documents, at least one of the synsets corresponding to the possible meaning of the word had to match. The matches were then normalised by the total number of synsets in both suspicious and source documents, based on the Jaccard coefficient (Formula 4.7 on page 108).

The results from dependency relation extraction (Feature 6) were calculated by extracting the unique relations from the output, and then the matches were normalised by the total number of syntactic dependency relations in both suspicious and source documents, again based on the Jaccard coefficient (Formula 4.9 on page 109).

6.1.4 Document Filtering

In plagiarism detection tasks, it is essential to perform initial filtering with superficial techniques to reduce the number of potential document pairs before the detailed analysis stage. The filtering stage allows the application of deeper NLP techniques such as syntactic and semantic analysis to a subset of the documents. In this experiment 5-gram matching (Formula 4.3 on page 104) on Feature 4 was chosen as the filtering metric, as it was efficient.

After using overlapping 5-grams to compute similarity between 814 suspicious documents and 10,416 source documents, the comparison results were sorted according to the total number of matching 5-grams. The top 10 values were then selected from all potential document pairs. The document pairs which did not have at least 10 common 5-grams or which had a very low similarity score were removed from the set. The selected 1,534 candidate document pairs were then passed on to the further processing stage with deep NLP techniques, which included lexical generalisation (Feature 5) and dependency relations extraction (Feature 6).

The final stage was to treat the problem as a classification task. Thresholds were established for the features in order to determine which document pairs were considered as plagiarised. Various thresholds were tested by plotting the change of threshold against their effects on classification performance. The results for the 1,534 candidate document pairs are shown in the next section.

6.1.5 Results

Features 1, 2, 3 and 4 were tested as the comparative baselines and the best baseline was selected for the final comparison. After evaluating the performance of the four n-grams features, Feature 4 was selected as the comparative baseline to measure against other features.

The use of 5-grams within sentence boundaries (Feature 3) and 5-grams across sentence boundaries (Feature 4) has effects on the classification of false positive cases. Feature 3 incorrectly identified more false positive cases (238 cases) than Feature 4 (171 cases), but the number of true positive and false negative cases identified by the two features is very close. The details are available in Appendix C, Table C.1.

Since a binary document classification was adopted, cases below the pre-set thresholds were considered as non-plagiarism cases, and cases above the threshold were considered as plagiarism cases. The standard evaluation metrics of precision, recall and F-score were employed to measure the detection performance.

The similarity scores were then tested on various levels of threshold to analyse the variation in the detection performance. Figures 6.1, 6.2 and 6.3 show the precision, recall and F-score at various thresholds respectively.

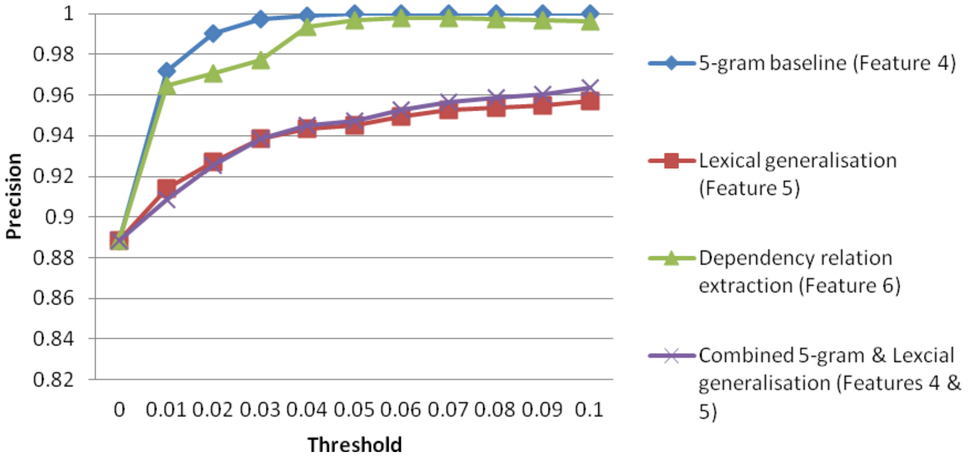

Figure 6.1: Precision for several thresholds in the similarity metrics

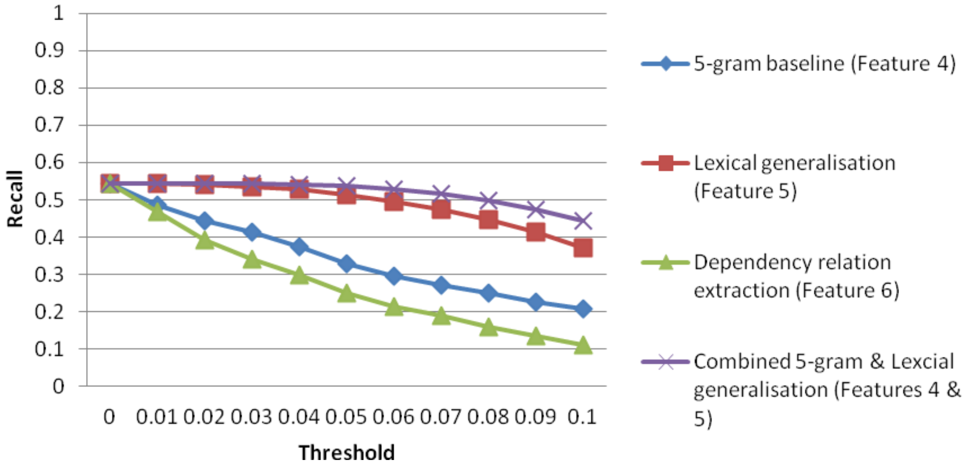

Figure 6.2: Recall for several thresholds in the similarity metrics

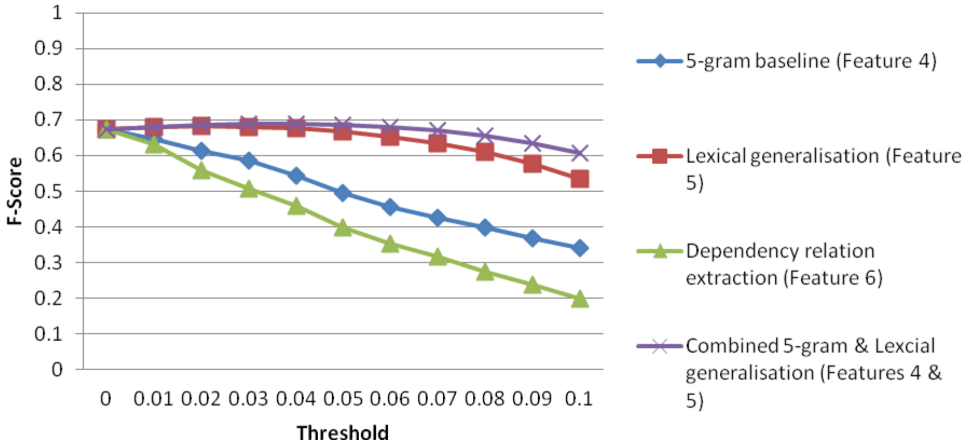

Figure 6.3: F-score for several thresholds in the similarity metrics

The results show that lexical generalisation was the best performing feature that matches with the baseline. Figure 6.4 shows the comparative performance between the 5-gram baseline (Feature 4) and lexical generalisation (Feature 5).

Table 6.3 shows the precision, recall and f-score for the selected features on the best performing threshold at 0.03, based on a binary classification.

Feature	Description	Precision	Recall	F-Score
4	5-gram baseline	**97.95%**	41.90%	58.69%
5	Lexical generalisation	93.83%	53.43%	68.09%
6	Dependency relations extraction	97.72%	34.20%	50.67%
4 & 5	**Combined 5-grams & Lexical generalisation**	93.85%	**54.23%**	**68.74%**

Table 6.3: Comparative performance of selected features

Upon further analysis at the individual levels of obfuscation, that is, the four levels of plagiarism annotation in the corpus (manual paraphrase, low artificial obfuscation, high artificial obfuscation, and no obfuscation), it is notable that

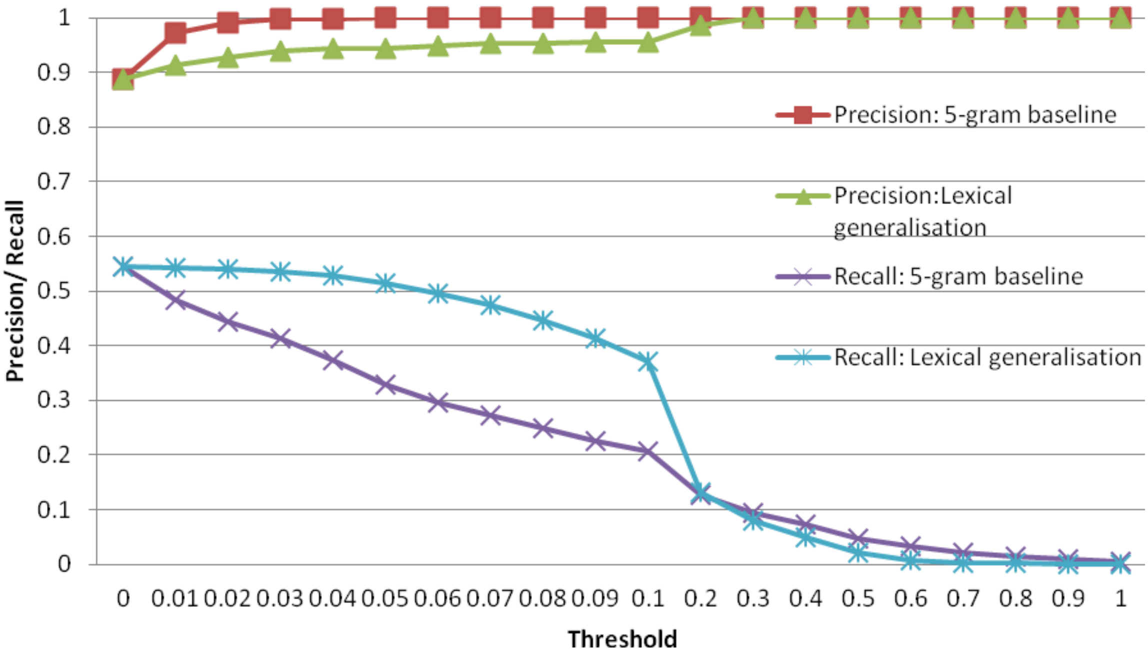

Figure 6.4: Comparative performance between the baseline (Feature 4) and lexical generalisation (Feature 5)

the use of lexical generalisation is more effective than the 5-gram baseline in all obfuscation levels. Although the baseline is effective in detecting direct verbatim copies, lexical generalisation is capable of achieving better results regardless of how the plagiarised texts were produced. In particular, this strategy identified significantly more simulated and obfuscated plagiarism cases than the baseline. Figure 6.5 shows the recall of both approaches on different levels of obfuscation, based on a threshold of 0.03.

The results from Features 1 and 2 were significantly outperformed by all other features, thus no further experiments were performed on these sets (see Appendix C Table C.1 for the list of results).

The results show that the 5-gram overlap metric performed at the sentence

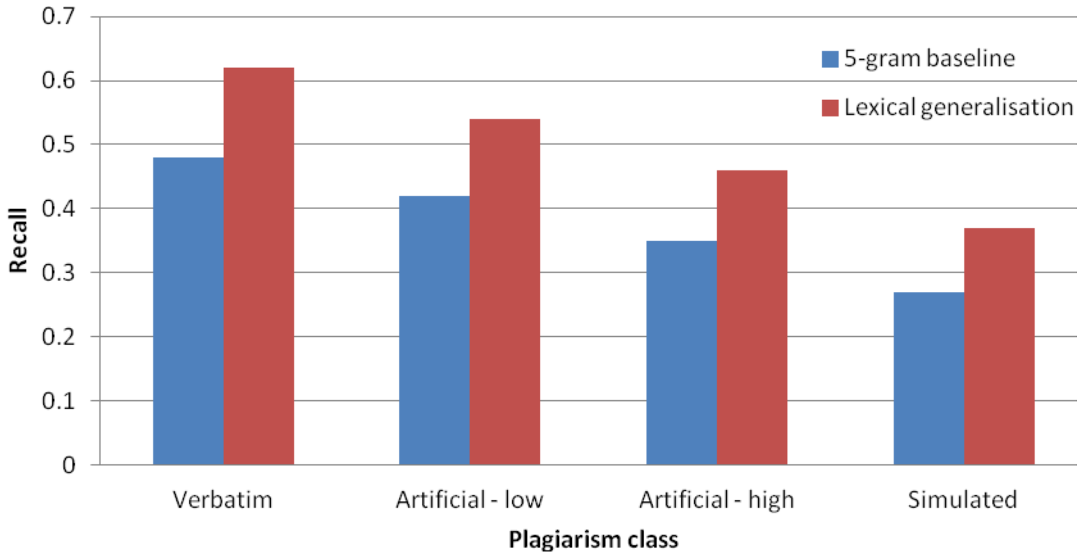

Figure 6.5: Recall obtained by the 5-gram overlap baseline and the synset-based similarity matching for different obfuscation levels

level (Feature 3) returned more false positive cases than its performance at the document level (Feature 4). As the number of true positive cases identified by both methods is similar, 5-gram document level was chosen in order to maintain a lower false positive rate.

The effects of applying different levels of thresholds on document classification were shown in Figures 6.1, 6.2 and 6.3. It is clear that the best overall performance was between the thresholds 0.01 and 0.02. Increasing the threshold can result in a higher precision but a reduction in recall, causing the F-score to drop. Even when the threshold is set at 0, the precision is above 0.88, recall is below 0.55 and F-score is near 0.65. This indicates that most false negative cases, which are the true plagiarism cases that we should have detected, were misidentified during the initial 5-gram overlap filtering stage.

The relatively high score in precision and low score in recall suggests that it is better to lower the thresholds in the filtering stages to expand the number of potential pairs to be examined in order to improve recall and F-score.

The results show that lexical generalisation (Feature 5) was the most promising technique, demonstrating over 0.91 precision and the best recall at all thresholds, with an F-score indicating an overall better performance than other methods. The high recall score shows that using all synsets in the comparison metric can help to reduce the number of false negative cases, that is, reducing the real cases of plagiarism that are not detected. However, the relatively low score in precision suggested that using all synsets may be too lenient. Therefore, lexical generalisation would be ideal if used to investigate a subset of highly suspicious plagiarism cases after filtering by other methods.

The results from 5-gram overlap matching at the document level (Feature 4) showed a balanced performance. The use of punctuation removal and stemming improved the precision but not the recall.

Results from dependency relation extraction (Feature 6) did not show a major improvement over other methods. This may be due to an issue with the parser settings, which truncate or omit very long sentences. The input documents were pre-processed with sentence segmentation which was based on using full stops to mark the end of sentence. This means that if a sentence has lots of commas it will be taken as a very long sentence until a full stop separates it. Also, in an attempt to speed up detection, the comparison metric did not consider the order of the dependency relations when computing the match, which may have affected the overall calculation of semantic relations.

Furthermore, the feature lexical generalisation (Feature 5) counts the number of matching synsets between texts without investigating the actual word relations. It may be interesting to incorporate word alignment metrics in order to reflect word relations between texts.

6.1.6 Discussion

For the purpose of evaluating the scalability of the proposed plagiarism detection framework and algorithms, which is the **third objective** of this thesis, this experiment was performed with a subset of the corpus at the document-level. In order to optimise detection performance, additional investigations are needed to seek a better similarity metric and a more accurate filtering technique, and to incorporate other structural features analysis techniques. Moreover, approaches from authorship attribution and word alignment from machine translation may provide possible improvements to the current detection framework.

Further analysis showed that using the 5-gram overlap metric is effective in detecting direct copies, while the use of lexical generalisation is effective in detecting obfuscated plagiarism cases. By using a combination of the two approaches, a slight improvement on the detection performance could be observed, but more experiments are needed to confirm the findings. It may be more effective to incorporate a machine learning algorithm to classify the documents based on similarity scores generated from different metrics (such as the lexical-based and semantic-based metrics), as shown in the small-scale experiment in the previous chapter.

This relates to the **second objective** of the thesis, which is to investigate the role of machine learning in our framework.

This experiment has shown the influence of deep NLP techniques on plagiarism detection performance. It suggests that using lexical generalisation can improve overall classification performance. Various levels of threshold have different effects on precision, recall and F-score; a lower threshold allows more cases to be investigated, whilst a higher threshold provides fewer cases but they are more likely to be plagiarised. Therefore, the threshold needs to be set in accordance with the detection task requirements. The filtering of potential documents and process efficiency remain as issues to be examined in future.

Further investigation into semantic parsing by using semantic role labellers can provide deeper analysis in terms of the semantic structure of texts. It is expected that semantic parsing, which gives rich features, will be more effective in identifying simulated plagiarism cases.

6.2 Document-level Additional Experiment

This section describes the additional experiment performed on a subset of the PAN-PC-10 corpus at the document level, using binary classification on manually simulated plagiarised documents. Three additional deep NLP techniques, including predicate extraction, predicate generalisation, and named entity recognition were investigated in this section.

6.2.1 Corpus

The corpus used in this experiment consists of the simulated cases from the PAN-PC-10 corpus. In the corpus, plagiarism cases are referred to as segments in suspicious documents, annotated in terms of character offsets. For the purpose of a document-level analysis, plagiarism cases are treated at the document level when a segment of a document pair returns a match.

In order to apply deep NLP techniques, it was essential to use grammatically and syntactically well-formed texts as input. The artificially generated cases were not suitable for use. 6% of the plagiarism cases in the PAN-PC-10 corpus are simulated plagiarism, where texts were manually rewritten. Consequently, the suspicious documents which contain manually simulated plagiarism are selected as the test data for this section, as shown in Table 6.4.

Document class	**Attribute**	**Statistics**
Source document	Number of documents	11,084
	Minimum document length	48 words
	Maximum document length	434,777 words
Suspicious document	Number of documents	903
	Minimum document length	499 words
	Maximum document length	70,500 words

Table 6.4: Corpus statistics

Since the goal is to investigate external plagiarism of English texts, all intrinsic and translated plagiarised documents were excluded from the dataset. The non-English source documents were translated into English automatically. The experiments presented in this section were based on 903 manually simulated plagiarised documents and 11,084 source documents, which gave a total of 10,008,852 possible pair-wise document-level comparisons.

A binary classifier was employed to classify each suspicious-source document pair as *plagiarised* or *clean*. Cases were treated at the document level rather than the passage level, where a pair of documents is considered as *plagiarised* whenever at least one segment within the suspicious document is plagiarised from the source document. Although in the PAN competition plagiarised cases are expected to be reported at the passage level, flagging plagiarised documents can already be very helpful for humans checking potential plagiarism cases by filtering out a very large percentage of documents from the process. Moreover, given that NLP techniques are much more computationally expensive than simple string matching techniques, document-level processing is a more realistic scenario for this feasibility study.

6.2.2 Text Pre-processing and NLP Techniques

The experiment followed the five-stage framework described in Section 4.1. The first three stages - pre-processing, similarity comparison and filtering - contribute to **candidate document retrieval**, that is, a filtering of documents in order to narrow down the search span of document pairs. The next two stages use deep NLP techniques to provide detailed analysis of the remaining candidate documents.

In order to generalise the texts for subsequent similarity comparisons, both source and suspicious documents were processed using text pre-processing and NLP processing techniques. The most promising techniques were again investigated in this section. The text processing techniques included tokenisation, lowercasing and punctuation removal. Shallow NLP techniques included stemming. Deep NLP techniques included lexical generalisation, dependency relation extraction, predicate extraction, predicate generalisation and named entity recognition.

To investigate the grammatical components of a sentence, all verbs were extracted from each document and the number of overlapping verbs are compared between text to form the feature predicate extraction. For predicate generalisation, the verbs extracted from predicate extraction were generalised using VerbNet by replacing the verbs by their respective VerbClasses.

To extract information such as names of persons, organisations and locations, named entity recognition was applied to extract named entities from texts.

6.2.3 Similarity Metrics

The selected corpus was processed using the following techniques and metrics shown in Table 6.5.

5-grams of words (Feature 1) was chosen as the baseline and filtering metric. Additional deep NLP techniques (Features 4, 5 and 6) were new to the framework. They were applied with their corresponding overlap coefficient metrics to calculate the number of similarities between document pairs. Predicate extraction (Feature 4) investigates the number of common verbs between documents, and predicate generalisation (Feature 5) investigates the number of common verb senses, which is a similar technique to lexical generalisation. Named entity recognition (Feature 6) investigates the number of common named entities between documents.

Feature	Techniques	Similarity Metric(Formula)
1	Tokenisation, Lowercasing, Punctuation removal, Stemming, 5-grams of words	Overlap coefficient (Formula 4.3, p.104)
2	Tokenisation, Lowercasing, Punctuation removal, Stopword removal, Lexical generalisation	Lexical generalisation (Formula 4.7, p.108)
3	Sentence segmentation, Dependency relation extraction	Dependency relation extraction (Formula 4.9, p.109)
4	Sentence segmentation, Part-of-speech tagging, Predicate extraction	Predicate extraction (Formula 4.10, p.109)
5	Predicate extraction, Lowercasing, Lemmatisation, Predicate generalisation	Predicate generalisation (Formula 4.11, p.110)
6	Sentence segmentation, Part-of-speech tagging, Named entity recognition	Named entity recognition (Formula 4.12, p.110)

Table 6.5: Dataset and techniques

6.2.4 Document Filtering

The use of progressive filtering makes the application of deep NLP techniques such as syntactic and semantic analysis more feasible in the remaining document pairs. The filtering stage is referred to as the candidate document retrieval, where document pairs that have high probability of plagiarism are referred to as candidate documents.

The 5-gram overlap (Feature 1) was selected as the filtering metric. Similarity scores were generated using the overlap coefficient (Formula 4.3 on page 104) by comparing each suspicious document against the whole source document collection. For each suspicious document, the top five ranked source documents with the highest similarity scores were selected as candidate document pairs. This gave

4,515 candidate document pairs which were treated as positive cases at this stage. The remaining 10,004,337 pairs were all treated as negative cases. The candidate document pairs were then passed to the next stage for further processing.

The features generated by the similarity metrics were passed to a machine learning classifier for training and testing. The InfoGain attribute evaluator was applied to identify the most contributing features. Then, the C4.5[29] (Formula 4.16 on page 116) 10-fold cross-validation classifier was applied on the 4,515 document pairs.

6.2.5 Results

The experiment adopted a binary classification for document pairs, where a pair is classed as *plagiarised* when its features reached a certain threshold, or *clean* otherwise. The standard evaluation metrics of precision, recall and F-score were employed to measure the detection performance.

After filtering, the 4,515 candidate document pairs contained 999 plagiarised cases and 3,516 clean cases; the filtering process missed 372 plagiarised cases. Based on these numbers, the performance of the filtering stage has a precision of 0.22, recall of 0.73 and F-score of 0.34. Ideally, a detection approach should make sure that all potential document pairs are flagged (high recall), but also make sure that clean documents are not flagged (high precision). This is to reduce the amount of human resources needed when manual analysis is required for the flagged documents. However, as in most classification tasks, high recall may come

[29] The Weka implementation of the C4.5 algorithm, J48, was used in this experiment.

at the price of low precision, and vice-versa. Therefore, depending on the detection task, it may be more important to drop one metric in favour of another.

The 4,515 document pairs were tested in the next stage, classification, using features generated by various techniques and similarity metrics listed in Table 6.5. The similarity scores for each document pair from each feature were then trained and tested with the machine learning classifier using C4.5 10-fold cross validation. The accuracy of individual features as well as combined features is shown in Table 6.6. A detailed list of results including the precision, recall and F-score of the two classes is available in Appendix C Table C.2.

Feature	**Description**	**Accuracy**
1	5-grams baseline	91.52%
2	Lexical generalisation	80.75%
3	Dependency relation extraction	77.87%
4	Predicate extraction	80.20%
5	Named entity recognition	83.70%
6	Predicate generalisation	79.40%
Combined	Features 1 & 2 & 4 & 5 & 6	91.58%
Combined	Features 1 & 2 & 4 & 5	91.61%
Combined	Features 1 & 2 & 4 & 6	91.63%
Combined	**All features**	**91.65%**

Table 6.6: Accuracy of individual features and combined features

The results show that the 5-grams metric achieved good performance, and individual deep NLP features did not match the performance of this baseline. However, the use of combined features brought a slight improvement.

The results suggest that more document pairs should be selected for further processing instead of just selecting the top five, which could help to further improve the recall. The low precision indicates that a balancing threshold needs to be set in order to reduce the number of false positives. Table 6.7 shows the comparison between the baseline features and all features.

Feature	**Clean**			**Plagiarised**		
	Precision	Recall	F-score	Precision	Recall	F-score
5-gram baseline (Feature 1)	94.2%	**95%**	94.6%	**81.8%**	79.4%	80.5%
All features	**94.6%**	94.7%	94.6%	81.2%	**81.1%**	**81.1%**

Table 6.7: Comparison of the baseline and all features based on two document classes

Although the combined features showed a slight improvement in the evaluation against other approaches, the contribution of individual NLP techniques in detection performance needs to be investigated. It is observed that a trend of using a combined approach is effective in improving the recall of document identification in the plagiarism class, which could be very useful as a filtering step to select candidate documents for in-depth passage-level investigation.

Dependency relation extraction performed below expectation in this experiment, and the same trend was observed in the previous experiment. The performance of other deep NLP techniques was below expectation too, and this is because these techniques require the actual plagiarised text segments to be isolated in order to investigate the deeper linguistic structure. Applying such techniques on either document was not favourable to the experiment, and overlapping n-grams was more consistent in this case.

6.2.6 Discussion

The results showed that the use of overlapping n-grams could be an effective filtering metric, and that deep techniques are better reserved for the detailed analysis stage. A better filtering strategy is required to optimise the detection performance. Instead of using the top five candidate document pairs, the subsequent experiment will investigate the use of the top ten or even more document pairs in the initial filtering stage in order to reduce the number of false negative cases. Approaches such as word alignment used in Statistical Machine Translation (SMT) and stylistic techniques used in authorship attribution may provide additional improvements.

An interesting note is that even though the experiment was exhaustively performed on simulated cases only, the overall improvement is not at all significant. Although NLP techniques are very useful in the analysis of linguistic information, they are based on the assumption that the input text must follow syntactic and grammatical rules. Investigating simulated cases at the document level is not ideal as the document does not only contain the manually rewritten text, but also other text that has been added to make up a complete document. These "filler texts" are noisy data that diminished the benefits of applying deep linguistic techniques. Hence, in this experiment the application of deep NLP techniques is not beneficial even if simulated documents were selected. Further investigation should be based on the actual segment for the plagiarised text.

The above reason intuitively set the groundwork for the next experiment. Further experiments using the PAN corpus are performed at passage level instead of document level in order to fully explore the role of NLP in plagiarism detection. Passage-level detection also allows a comparative evaluation against other PAN competitors.

6.3 Passage-level Experiment

This section describes the passage-level experiment performed on a subset of the PAN-PC-10 corpus. Comparative evaluation is performed with two other PAN competitors. Different from previous experiments, this section treats plagiarism cases at the passage level; passages are extracted from the documents, and all passages are plagiarised texts. The manually simulated plagiarism passages are used, where writers attempt to cover their trails by synonym substitution, change of sentence structure and paraphrasing. The task is to identify which plagiarised passage corresponds to a particular original passage. In other words, a plagiarised-source passage pair that display similar content is treated as plagiarism; whereas a pair that does not resemble each other is treated as clean. Evaluation is performed with a binary classification of *plagiarised* or *clean* at the passage level between plagiarised-source text pairs, and compared against an overlapping n-gram baseline.

6.3.1 Corpus

Similar to the previous section, the corpus used in this experiment also consists of the simulated cases from the PAN-PC-10 corpus. As mentioned previously, the PAN-PC-10 corpus referred to plagiarism cases as segments in suspicious documents. The difference here is that instead of using the simulated cases as a document, plagiarism passages were extracted from the aforementioned documents. In other words, the experiment is based on the hypothesis that the candidate documents had been selected and had reached the detailed analysis stage. Performing experiment at the passage-level removes the need to compare the difference in the pre-processing stage between various detection systems. If there is no need to extract text segments from the corpus, it will allow the actual system effectiveness to be measured.

The previous experiment has shown that the availability of grammatically and syntactically well-formed texts is needed for the application of deep NLP techniques. In this section, the passage level subset contains only the manually rewritten text segments. The plagiarised passages along with the original source passages were extracted according to the corpus annotation. Table 6.8 shows the corpus statistics. The corpus used in this experiment contains 4,067 suspicious passages and 4,067 source passages, which gives a possible 16,540,489 pair-wise comparison.

Passage class	**Attribute**	**Statistics**
Source passage	Number of passages	4,067
	Minimum length	28 words
	Maximum length	954 words
	Average length	491 words
Plagiarised passage	Number of passages	4,067
	Minimum length	19 words
	Maximum length	1,190 words
	Average length	604.5 words

Table 6.8: Corpus statistics

6.3.2 Text Pre-processing and NLP Techniques

The experiment followed the five-stage approach described in Chapter 4. The first stage, pre-processing, involves processing the passages with text pre-processing and shallow NLP techniques. Techniques that showed the most promising results from the previous experiments were applied, including shallow techniques such as stemming. For this experiment, further processing (Stage 4) using deep NLP techniques such as lexical generalisation and dependency relation extraction was applied as part of Stage 1. This is because the focus of the detection framework has now been given greater weight in the detailed comparison stage instead of in the filtering stage. In previous experiments, deep techniques were applied only to the selected candidate pairs as they are computationally expensive to apply in large document collections. However, the experimental settings in this section allowed the application of deep techniques alongside shallow techniques as the average case length is much shorter. Therefore, it was feasible to apply deep NLP techniques as part of the pre-processing stage. In addition, this section explored an NLP technique that had not been investigated before: word alignment.

Overlapping 3-grams of words was chosen as the evaluation metric, as this corpus has shorter case length. The reason for using 3-grams instead of 5-grams was because 3-grams handles shorter case length more effectively than 5-grams, where 5-grams was used in previous experiments with longer document length. Feature 1, overlapping 3-grams, was used as the baseline of this experiment.

To investigate the relations between words, Feature 7 was based on word alignment using the METEOR tool,[30] which performs pre-processing and ranking automatically. For each suspicious-source pair, the words are aligned based on 1) exact words, 2) stemmed words, 3) synonyms and 4) paraphrases, with different weights given to each of the four word-matching categories.

Similarity metrics were then applied to the processed texts to generate similarity scores for each suspicious-source passage pair. The scores represent features that were investigated individually and as various combinations in the classification stage.

6.3.3 Similarity Metrics

After applying the techniques, all suspicious-source pairs were passed to the similarity metrics to compute the similarity scores. The 3-grams baseline was calculated by the overlap coefficient (Formula 4.3 on page 104), where the unique 3-grams in the case pair were normalised by the shorter case. As well as being a feature, overlapping 3-grams (Feature 1) also contributes as a filtering strategy. For each of the plagiarised passages, the top ten source cases ranked by the overlap similarity score were selected as the candidate cases. This resulted in 40,300 possible case pairs, and cases that have 0% similarity were eliminated. The candidate cases were then extracted for similarity score calculations for other features. Table 6.9 shows the combination of techniques and similarity metrics. The similarity scores for each pair were then used as features in the machine learning classification.

[30] http://www.cs.cmu.edu/~alavie/METEOR/

Feature	Techniques	Similarity Metric(Formula)
1	Tokenisation, Lowercasing, Punctuation removal, Stemming, 3-grams of words	Overlap coefficient (Formula 4.3, p.104)
2	Tokenisation, Lowercasing, Punctuation removal, Stopword removal, Lexical generalisation	Lexical generalisation (Formula 4.7, p.108)
3	Sentence segmentation, Dependency relation extraction	Dependency relation extraction (Formula 4.9, p.109)
4	Tokenisation, POS-tagging, Predicate extraction	Predicate extraction (Formula 4.10, p.109)
5	Tokenisation, POS-tagging, Lowercasing, Lemmatisation, Predicate generalisation	Predicate generalisation (Formula 4.11, p.110)
6	Sentence segmentation, Part-of-speech tagging, Named entity recognition	Named entity recognition (Formula 4.12, p.110)
7	Word alignment	—

Table 6.9: Text processing and NLP techniques used in passage-level processing

6.3.4 Machine Learning Classification

A decision tree classifier was used to classify each suspicious-source passage pair into a class, based on individual or combined features. The classifier used was decision tree algorithm C4.5[31](Formula 4.16 on page 116), which was applied to the 40,300 passage pairs. For a comparative evaluation, the 40,300 candidate instances (suspicious-source cases) were split into 3 groups. For the classification task, 2/3 of the instances (i.e. two splits) were used as the training data and tested with the remaining 1/3 (i.e. one split) testing data.

The InfoGain attribute evaluator (Formula 4.14 on page 114) was also applied to identify the most contributing features. The "Best Features Set" was a combina-

[31] The machine learning algorithm used is the WEKA implementation of C4.5, J48 classifier.

tion of the best performing techniques ranked by the InfoGain attribute evaluator, which included the most contributing features: dependency relation extraction (Feature 3), word alignment (Feature 7), overlapping 3-grams (Feature 1), named entity recognition (Feature 6), and lexical generalisation (Feature 2).

For the evaluation of individual features, additional training and testing is based on a 10-fold cross-validation using the decision tree classifier.

6.3.5 Results

The results obtained by the decision tree 10-fold cross-validation on individual features are listed in Table 6.10. The full list of the results is available in Appendix C Table C.3.

Feature	**Clean**			**Plagiarised**			**Accuracy**
	Precision	Recall	F-score	Precision	Recall	F-score	
1	98.2%	98.8%	98.5%	87.7%	83%	85.3%	97.26%
2	96.4%	97.7%	97%	74.9%	65.8%	70%	94.61%
3	**98.8%**	98.7%	**98.7%**	79%	**88.4%**	**88.1%**	**97.72%**
4	93%	**99.1%**	95.9%	76.9%	29.6%	42.7%	92.41%
5	95.5%	98.7%	97.1%	82.3%	56%	66.6%	94.63%
6	97.3%	98.9%	98.1%	87.5%	73.9%	80.1%	96.49%
7	98.2%	**99.1%**	**98.7%**	**91%**	82.6%	86.6%	97.55%

Table 6.10: Results of individual features

The results show that the 3-grams baseline itself was already an effective feature on its own, as no other individual features perform as well. However, when combining the best performing features as the Best Features Set, the set outperformed all individual features, which is shown in the comparative analysis.

An extra experiment was conducted to investigate the impact of machine learning on the classification of passage level plagiarism. It was noted that the machine learning performed on a single feature did not show significant improvement over the string-matching method. The details can be found in Appendix C Table C.4.

A comparative evaluation of the results of this experiment was performed against two other best-performing PAN competitors. Two of the PAN competitors, Muhr et al. (2010) and Grozea and Popescu (2010), applied their approaches to the same test cases used in this experiment. Muhr et al. (henceforth referred to as PAN1) ranked fifth out of 13 teams in the PAN-PC-09 task, and third out of 18 teams in the PAN-PC-10 task. Grozea & Popescu (henceforth referred to as PAN2) ranked first in the PAN-PC-09 task and fourth in the PAN-PC-10 task. Machine learning was not included in their systems, and they performed standard text pre-processing techniques that included tokenisation, lowercasing and punctuation removal. No other NLP techniques were used and the methods were based on n-gram comparison. Both research teams calibrated their methods to accommodate the shorter text length in the corpus, as the main focus of this experiment is not to establish the exact location of the plagiarised text segments, but the effectiveness in identification of paraphrased texts.

PAN1 employs a two-stage detection approach. The first stage is to search for matching plagiarised text blocks within the overlapping source document blocks, with each source block containing 40 words and the suspicious documents split into overlapping blocks of 16 words. The second stage is to post-process matching blocks to calculate the locations of the plagiarised texts. Once a suspicious-source block has reached a pre-determined threshold, the blocks are then filtered by Jaccard similarity. For the purpose of this experiment, the authors fine-tuned the approach to handle shorter cases, thus less emphasis was placed on the second stage as the initial similarity filtering was already competent at finding the most matching pairs.

PAN2 also employs a two-stage detection approach. The first stage is to filter by pair-wise matching between suspicious and source cases based on character 16-grams. The pairs with the highest similarity scores are further investigated in the second detailed analysis stage. In the original work described in Grozea et al. (2009) the second stage of the detection approach involved the computation of the exact locations of the plagiarised text in large documents. Similar to PAN1, the authors calibrated the system to adapt to a shorter case length for the purpose of this experiment. As a result, less emphasis was placed on the second post-processing stage.

To perform a comparative evaluation, the results obtained from their systems were split in the same way as in the previous section. Then the standard evaluation metrics of precision, recall, f-score and accuracy were applied. Statistical significance using the z-test at a confidence level of 95% was observed in the recall and f-score o the plagiarism class. A detailed list of results is available in Appendix C Table C.5. A comparison between the performances of different detection approaches is listed in Tables 6.11 and 6.12. Statistically significant performance is observed in the recall of the plagiarised class using the Best Features Set.

Statistical significance is calculated with the z-test for proportions using depen-

Class	Feature	Precision	Recall	F-score
Plag	Baseline	0.869 ± 0.008	0.838 ± 0.009	0.854 ± 0.007
	Best Features	0.901 ± 0.014	**0.952±0.004**	**0.926±0.006**
	PAN1	0.905 ± 0.016	0.799 ± 0.017	0.848 ± 0.005
	PAN2	**0.953 ± 0.005**	0.757 ± 0.005	0.844 ± 0.004
Clean	Baseline	0.983 ± 0.001	0.987 ± 0.001	0.985 ± 0.001
	Best Features	**0.995 ± 0.001**	0.989 ± 0.002	**0.992 ± 0.001**
	PAN1	0.979 ± 0.002	0.991 ± 0.002	0.985 ± 0
	PAN2	0.975 ± 0	**0.996 ± 0.002**	0.985 ± 0

Table 6.11: Average precision, recall, and F-score with standard deviation for the experiment

Feature	Accuracy
Baseline	0.973 ± 0.001
Best Features	**0.985 ± 0.001**
PAN1	0.973 ± 0.0003
PAN2	0.973 ± 0.0003

Table 6.12: Accuracy with standard deviation for the experiment

dent groups (i.e. all cases are the same in each group). P = 0.05 at a confidence level of 95%. Significance was observed for all splits.

Statistical significance was not observed between the Best Features Set and the two PAN competitors in the clean class. Interestingly, we found that the Best Features were better at reducing the false negative cases, as the Best Features Set outperformed all other comparative features and the baseline in the plagiarised class in terms of recall. This is shown in Figure 6.6, where the Best Features either matched or outperformed all the other features in both plagiarised and clean classes.

Simulated cases examples excerpted from the PAN-PC-10 corpus are shown below. Needless to say, manually simulated cases are more linguistically coherent

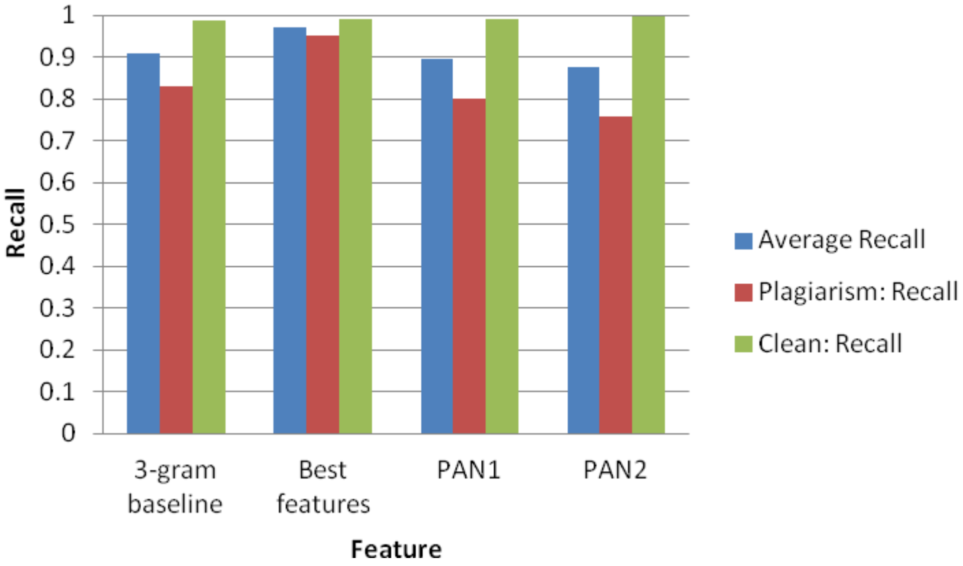

Figure 6.6: Detailed comparison of recall

than artificially generated cases. It is observed that the simulated cases retained the original sentence and grammatical structure of the source text to a greater extent. This inspired the application of translationese theories in the detection of plagiarism direction, which is described in Chapter 7.

Example Source 1 "M. Comte would not advise so irrational a proceeding. But M. Comte has himself a constructive doctrine; M. Comte will give us in exchange–what? The Scientific Method! We have just seen something of this scientific method."

Plagiarised 1 "Even M. Comte would spurn such irrational reasoning. However, M. Comte adheres himself to a fruitful belief, one which he will offer us instead - the Scientific Method! This scientific method has, in fact, just been observed."

Example Source 2 “Without enumerating all the modern authors who hold this view, we will quote a work which has just appeared with the imprimatur of Father Lepidi, the Master of the Sacred Palace, in which we find the two following theses proved: 1.”

Plagiarised 2 “Just without specifying the current writers who have this view, we will proceed with the work just came with the imprematurе of Father Lepidi, the Master of Sacred palace, which proves the following theses proved: 1.”

Example Source 3 “Therefore, a person should search his actions and repent his transgressions previous to the day of judgment. In the month of Elul (September) he should arouse himself to a consciousness of the dread justice awaiting all mankind.”

Plagiarised 3 “As such, a person should analyze what he did and be sorry for his mistakes before judgment day. In September, also referred to as Elul, he should force himself of the frightening justice that awaits all humans.”

Example Source 4 “I have heard many accounts of him, said Emily, all differing from each other: I think, however, that the generality of people rather incline to Mrs. Dalton’s opinion than to yours, Lady Margaret. I can easily believe it.”

Plagiarised 4 “Emily said, I have heard many different things about him; however, most people trust Mrs. Dalton’s beliefs more then they do yours, Lady Margaret, myself included.”

Table 6.13 shows the results obtained using various features and the two PAN competitors on the example cases.

E.g.	**Feature**							**PAN1**	**PAN2**
	1	**2**	**3**	**4**	**5**	**6**	**7**		
1	0.088	0.352	0.103	0.25	0.167	1	0.280	-0.532	0
2	0.235	0.733	0.273	0.286	0.625	0.667	0.371	0.919	1
3	0.029	0.346	0.069	0	0	1	0.273	-0.426	0
4	0.16	0.434	0.25	0.5	0.8	1	0.243	-0.925	0

Table 6.13: Similarity scores of individual features and the two PAN systems

These raw results are extracted from the experiment and are not normalised. In Table 6.13, the features are investigated individually rather than combined with a machine learning model. PAN1 identifies plagiarised cases as positive numbers, and marks non-plagiarised cases as negative numbers. PAN2 simply marks plagiarised cases as 1 and non-plagiarised cases as 0.

It is noted that named entity recognition (Feature 6) performed particularly well. Interestingly, on case 2 the feature did not perform as well, which is due to the misspelling of the word “impremature” in the plagiarised document being counted as part of the named entity “Father Lepidi”.

In Table 6.14, the four examples were investigated with a machine learning classifier and the results showed that the best features outperformed the baseline and the two PAN systems, which correctly identified all four cases. TP stands for a correctly identified case, whereas FN stands for a plagiarised case incorrectly classified as clean.

One of the most contributing features identified by the InfoGain attribute evaluator is word alignment. The results confirmed the hypothesis that deep NLP

E.g.	**Baseline (Feature 1)**	**Best features (Features 1, 2, 3, 6 & 7)**	**PAN 1**	**PAN 2**
1	TP	TP	FN	FN
2	TP	TP	TP	TP
3	FN	TP	FN	FN
4	TP	TP	FN	FN

Table 6.14: Comparison of detection performance based on the example cases

techniques can help to improve the classification of plagiarised texts. Furthermore, it is shown that a simple string-matching algorithm (3-grams baseline) is already capable of identifying short plagiarised texts, provided the text maintains the same structure and use of words to a large extent, as did examples 1, 2 and 4. In scenarios where the texts have undergone substantial paraphrasing, such as example 3, deep NLP techniques combined with simple string-matching algorithms is an effective approach.

6.3.6 Discussion

This experiment has advanced from the document-level retrieval stage towards a separate passage-level detailed comparison stage. The experiment compared various approaches of plagiarism detection, from the basic string-matching approaches (overlapping n-grams of words and characters) to the latest information retrieval approaches used in the PAN competitions, against our proposed NLP-inspired framework. Evaluation has shown that the combination of string matching and NLP techniques is again the best performing approach. An interesting point to note is that dependency relation extraction did not show stable performance as it did in the previous small-scale experiment (Chapter 5). It is speculated that this is due to the difference in the nature of the corpus: namely, shorter text cases and paraphrasing with word replacements. Rather than seeing this as a flaw in the technique itself, further work will consider alternative ways to improve the application, analysis and evaluation of linguistic techniques (see Chapter 7).

The proposed approach showed significant improvement over other approaches, and the contribution of individual NLP techniques in detection performance was investigated. The combined approach is effective in identifying passage-level plagiarism cases and reducing false negative cases, which could be very useful in the detailed detection stage. An ideal detection approach should make sure that all potential plagiarised cases are flagged (high recall), but also make sure that non-plagiarised cases are not flagged (high precision), to save the amount of human resources needed for manually analysing the flagged cases.

This section demonstrated the influence of lexical, syntactic and semantic techniques on plagiarism detection performance as a binary classification task. In particular, the use of named entity recognition and word alignment showed promising results in improving the recall in the plagiarised class. It is shown that the use of a machine learning decision tree algorithms can achieve a better accuracy than statistical techniques without machine learning. This is due to the capabilities of machine learning for handling multiple features and a higher tolerance of noise in the dataset.

6.4 Summary

This chapter described the use of NLP techniques with similarity metrics to improve the performance of string-matching plagiarism detection approaches. The experiments were performed with three subsets of the PAN-PC-10 corpus.

Overall, this chapter demonstrated the influence of lexical, syntactic and semantic techniques on plagiarism detection performance in document-level and passage-level classifications. This met the **third objective** of the thesis which is to investigate the scalability of the proposed framework. Text processing and NLP techniques, especially the use of lexical generalisation, named entity recognition and word alignment, are promising techniques in plagiarised document classification tasks. The chapter also investigated the application of machine learning techniques to plagiarism detection, which achieved the **second objective** of the thesis, which is to investigate the role of machine learning in the framework. It is believed that the use of machine learning techniques can achieve a better accuracy than statistical techniques, as they are capable of handling multiple features and are more tolerant of noise in datasets.

The first experiment (Section 6.1) showed that a deep NLP technique, lexical generalisation, helped to improve the recall of plagiarised document classification at the document level. The follow-up experiment (Section 6.2) showed that combining string-matching metrics with deep NLP techniques helped to improve the recall of plagiarised document classification. The final experiment (Section 6.3) took it further by concluding that the combination of string-matching metrics and deep NLP techniques with machine learning can improve the recall of classification on plagiarised text passages. The combined approach also outperformed other PAN systems which are calibrated to be tested at passage level. This suggests that deep NLP techniques should be given more attention in the plagiarism detection field.

All three experiments confirmed the hypothesis that deep NLP techniques can improve the identification of plagiarised text. Although the deep NLP techniques did not bring substantial improvement to all parts of the classification task, it is apparent that a trend can be observed - that these techniques help to reduce false negatives.

To conclude, although deep NLP techniques come with a cost of high processing resources, the improvement gained by applying such techniques may be worthwhile to enhance simple string-matching metrics for a better result. Depending on the requirement of the plagiarism detection task, deep NLP techniques can be employed to favour recall. In order to optimise the performance of deep NLP techniques and to explore the role of linguistic information, further experiments are carried out and other approaches are investigated in the next chapter.

Chapter 7

Experiments with Plagiarism Direction Identification

This chapter describes the experiments to identify the direction of plagiarism, which are based on linguistically and statistically-inspired features to distinguish between plagiarised and non-plagiarised text. The aim is to train machine learning classifiers with morphological, syntactic and statistical features and investigate whether such features can be applied to the identification of plagiarism direction.

The identification of plagiarism direction is split into two main machine learning tasks: 1) to classify whether an individual case is plagiarised or original, and 2) to rank a pair of cases according to their "direction", that is, to determine which of the text in a pair of texts is derived from the other. An additional task is to perform a multiclass classification to determine if a piece of text belongs to one of the three classes: artificial plagiarism, simulated plagiarism or original.

In this chapter, Section 7.1 covers the corpus used, the proposed framework and experimental settings. Section 7.2 details the text pre-processing and NLP techniques. Section 7.3 describes the feature extraction and selection process. Section 7.4 lists the machine learning algorithms used for training and testing based on the selected features. Results are presented in Section 7.5 and discussion is presented in Section 7.6. The chapter concludes with a discussion in Section 7.7

that highlights the contribution of these experiments and addresses open questions.

7.1 Corpus and Framework

A supervised machine learning approach is proposed to test the hypothesis that original and plagiarised texts exhibit significant and measurable linguistic differences. The method is to train a model with various linguistically and statistically-motivated features and then test it with three sets of data. Different from previous work on plagiarism detection, the tests are performed at the passage level instead of the document level, and the features investigated are not based on brute-force string-matching metrics but instead on the fundamental linguistic differences between original and plagiarised texts.

The corpus used in this experiment consists of three distinct datasets: parallel simulated cases, parallel artificial cases and non-parallel artificial cases. Parallel cases are cases where the plagiarised texts and their associated original texts can be matched as a pair. This is different from the non-parallel cases where the plagiarised texts and the original texts are not associated. The datasets are extracted from the PAN-PC-10 plagiarism detection corpus(Potthast et al., 2010c), where simulated cases were manually rewritten via Mechanical Turk, and artificial cases were created via automatic means with two levels of obfuscation.

Some of the features used in the experiment were inspired by studies of translationese, Translation Universals and translation direction detection. Other features investigated include the use of statistical language models and syntactic tree kernels. The features are tested with two tasks: **classification task** and **ranking**

task. They are trained and tested with machine learning models where features are analysed with a rule-based classifier, a kernel-based classifier and a linear pairwise ranker.

Three datasets are extracted from the PAN-PC-10 corpus, as shown in Table 7.1.

Class	**Statistics**	**Parallel Simulated Dataset**	**Parallel Artificial Dataset**	**Non-parallel Artificial Dataset**
Original	Number of segments	4067	4000	4000
	Minimum length	74 words	46 words	46 words
	Maximum length	745 words	4506 words	4506 words
	Average length	409.5 words	2276 words	2276 words
Plagiarised	Number of segments	4067	4000	4000
	Minimum length	21 words	38 words	41 words
	Maximum length	1190 words	3917 words	4535 words
	Average length	605.5 words	1977.5 words	2288 words

Table 7.1: Corpus statistics

The simulated dataset is composed of all the manually paraphrased text segments from the corpus: 4,067 plagiarised cases and their corresponding 4,067 original texts.

The artificial plagiarism cases are mechanically generated using random text operations, which include replacing, shuffling, removing or inserting words at random. Another approach is to use semantic word variations that replace words by their synonyms, antonyms, hyponyms or hypernyms at random. The operations also include POS-preserving word shuffling that keeps the sequence of part-of-speech and shuffles the words at random.

The parallel artificial dataset is composed of a randomly selected set of 4,000 artificially generated highly obfuscated plagiarism cases and their corresponding 4,000 original texts. The non-parallel artificial dataset is composed of a randomly selected set of 4,000 artificially generated highly obfuscated plagiarised cases, and 4,000 original texts which are not aligned with the plagiarised texts. Only artificial cases that are highly obfuscated are used, as plagiarism cases with low obfuscation display very high similarities to the original texts, rendering them unsuitable for the directional detection experiment.

Following the plagiarism detection framework described in Chapter 4, a three-stage approach is employed for this experiment. The first stage is to generalise the corpus with text pre-processing, and shallow and deep NLP techniques. This is described in the next section. The second stage is to extract morphological, syntactic and statistical features from the corpus, as described in Section 7.3. The final stage is to classify or rank each case into its respective class, using machine learning models based on selected features, as explained in Section 7.4.

Stage 1: Pre-processing This stage prepares the input text collection, which includes both rewritten text segments and original text segments, with simple text pre-processing, and shallow and deep NLP techniques. This stage generalises the input data for subsequent stages.

Stage 2: Feature Extraction The morphological, syntactic and statistical traits are extracted and used as individual feature sets or as a combined feature set. The linguistically-inspired features are drawn from the studies of translationese, Translation Universals and translation direction detection (Section 3.4.4). The statistical features are based on the use of statistical language models.

Stage 3: Classification The final stage is to classify or rank each case into its respective class. This can be a classification task to classify each case into a class of Plagiarised or Source. Or this can be a ranking task to rank a plagiarised and source pair to see which version is most original. Classifications are verified by applying standard evaluation metrics which include precision, recall, F-score and accuracy.

The processing flow chart (Figure 7.1) shows the framework for plagiarism direction identification. A text collection pass through various stages of processing, and then features are extracted and selected to represent the rewriting traits of texts. The features are used in a text **classification** or **ranking** task before the evaluation.

The experiment of identifying parallel original and plagiarised texts is evaluated as two tasks, that is, **binary classification** and **pair-wise ranking**. Binary classification refers to the two classes, that is, *original* and *plagiarised*, and the goal is to assign all individual texts in the collection to their respective classes. On

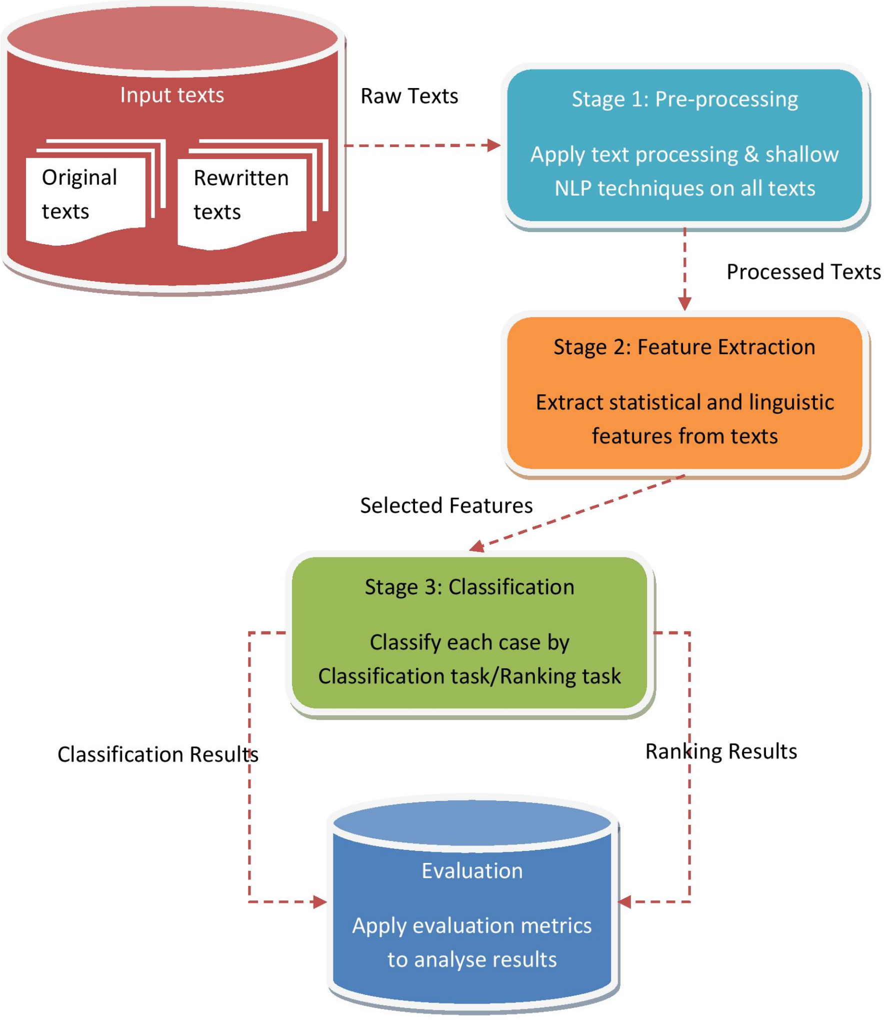
Input texts
Original texts
Rewritten texts
Raw Texts
Stage 1: Pre-processing
Apply text processing & shallow NLP techniques on all texts
Processed Texts
Stage 2: Feature Extraction
Extract statistical and linguistic features from texts
Selected Features
Stage 3: Classification
Classify each case by Classification task/Ranking task
Classification Results
Ranking Results
Evaluation
Apply evaluation metrics to analyse results

Figure 7.1: Plagiarism direction identification framework

the other hand, pair-wise ranking attempts to determine the direction of rewriting between a pair of parallel texts, that is, the ranker *sorts each pair of texts* to indicate which text has a higher level of rewriting. As the ranking task is capable of sorting multiple items, it could be further applied to identify multiple versions of rewritten texts, which is a difficult task for traditional binary classification.

The **multiclass classification task** is based on the parallel simulated and parallel artificial datasets, where 4,000 artificial plagiarism texts, 4,067 simulated plagiarism texts and 8,067 original texts are used. The task is to classify each text into one of the three classes: *artificial*, *simulated* or *original*.

The experiment with the non-parallel dataset, where the plagiarised texts and the original texts cannot be matched as a pair, is only treated as a classification task as the non-aligned cases cannot be ranked.

7.2 Text Pre-processing and NLP Techniques

To normalise the datasets for feature extraction and selection (Section 7.3), text pre-processing, and shallow and deep NLP techniques are applied. Shallow techniques include sentence segmentation, tokenisation, lowercasing, POS tagging and lemmatisation (see Section 4.2 for a detailed explanation of NLP techniques). Deep techniques include parsing, which generates the syntactic tree feature.

The techniques are applied in accordance with the requirements to extract features. The framework proposes a thorough investigation of morphological, syntactical, statistical and simplification features at both token level and sentence level. Our framework applies morphological and simplification features inspired by related studies on translationese, Translation Universals and translation direction, and also provides an in-depth study of each sub-category within the morphological features, such as using individual proportions of nouns, prepositions and pronouns, and individual function words as features (See Section 7.3 for a list of features).

For the morphological and simplification features, the following text pre-processing and shallow NLP techniques are applied:

1. Sentence segmentation
2. Tokenisation
3. Lowercasing
4. POS tagging
5. Lemmatisation

POS tags and lemmas of words are generated by the Stanford CoreNLP toolkit[32] (Klein and Manning, 2003; Toutanova et al., 2003). Following the study by Koppel and Ordan (2011), function words from a list (see Appendix E) are extracted as features. POS tagging is especially important as many of the morphological and simplification features depend on the POS tags.

The statistical features are pre-processed with sentence segmentation, tokenisation and lowercase. N-gram statistical language models are built using the KenLM [33] toolkit (Heafield, 2011) to calculate 1) log probability, 2) perplexity with all tokens, and 3) perplexity without the end-of-sentence marks. It is assumed that the 3-gram language model is better on shorter texts and the 5-gram language model is more suitable for longer texts. Therefore 3-gram and 5-gram models are adopted in order to provide a comparative analysis. The language models are trained with an in-domain corpus, which consists of 1.7 million original text segments from the PAN-PC-10 corpus that are not present in the testing datasets.

Finally, a syntactic feature should provide an additional linguistically motivated perspective for the experiment. Parsing is a deep NLP technique which dis-

[32] http://nlp.stanford.edu/software/corenlp.shtml

[33] http://kheafield.com/code/kenlm/

plays the structure of sentences in the format of phrase structure trees. Different from previous experiments described in Chapters 5 and 6 which used dependency relations, this experiment uses the actual syntactic trees for comparison. Syntactic information is extracted by generating syntactic trees using the Stanford PCFG parser (Klein and Manning, 2003), which forms part of the Stanford CoreNLP toolkit[34]. For example, for a plagiarised sentence "when you would have them red, you must cover them in the boyling." the following parse tree and dependency relations are generated:

```
(ROOT
  (S
    (SBAR
      (WHADVP (WRB when))
      (S
        (NP (PRP you))
        (VP (MD would)
          (VP (VB have)
            (S
              (NP (PRP them))
              (ADJP (JJ red)))))))
    (, ,)
    (NP (PRP you))
    (VP (MD must)
      (VP (VB cover)
        (NP (PRP them))
        (PP (IN in)
          (NP (DT the) (NN boyling)))))
    (. .)))
advmod(have-4, when-1)
nsubj(have-4, you-2)
aux(have-4, would-3)
dep(cover-10, have-4)
nsubj(red-6, them-5)
xcomp(have-4, red-6)
```

[34] http://nlp.stanford.edu/software/corenlp.shtml

```
nsubj(cover-10, you-8)
aux(cover-10, must-9)
dobj(cover-10, them-11)
det(boyling-14, the-13)
prep_in(cover-10, boyling-14)
```

The parse tree is then post-processed for analysis with syntactic tree kernels (see Section 7.3):

```
-1  (S (SBAR (WHADVP (WRB when))
(S (NP (PRP you)) (VP (MD would)
(VP (VB have) (S (NP (PRP them)) (ADJP (JJ red))))))) (, ,)
(NP (PRP you)) (VP (MD must) (VP (VB cover) (NP (PRP them))
(PP (IN in) (NP (DT the) (NN boyling))))) (. .))
```

If a segment is an original segment, it is marked as +1, whereas if a segment is a plagiarised segment, it is marked as -1.

After the techniques are applied, features are extracted and selected according to the morphological, simplification, syntactic and statistical traits.

7.3 Feature Extraction and Selection

To facilitate machine learning classification and ranking, features that capture the simplification, morphological, statistical and syntactical aspects of texts are investigated. These features should reflect the frequencies of the linguistic components of the texts.

In this experiment, the focus is on the use of simplification features. This is inspired by the simplification universal discussed in studies on translationese and Translation Universals, which suggests that translated texts use simpler and shorter words (Pastor et al., 2008; Mitkov and Pastor, 2008). This led us to consider the possibility that plagiarised texts may also use simpler and shorter words, which is investigated by extracting the following simplification features:

1. Average token length. In this study, the term "token" refers to word tokens. This is the number of characters normalised by the number of tokens.
2. Average sentence length. This is the proportion of number of word tokens per sentence, aiming to capture the shorter sentence length in plagiarised texts caused by splitting sentences.
3. Information load. This is the proportion of lexical words to tokens. Lexical words are represented by nouns, verbs, adjectives, adverbs and numerals.
4. Lexical variety. This refers to the type/token rate, by normalising the unique word type over all words.
5. Lexical richness. This is the proportion of type lemma per tokens. Different from lexical variety, lexical richness measures the lemmatised word type normalised by all words.
6. Proportion of sentences without finite verbs.
7. Proportion of simple sentences. Simple sentence refers to a sentence that contains only one finite verb.
8. Proportion of complex sentences. Complex sentence refers to a sentence that contains more than one finite verb.

To capture plagiarised traits that may occur at a morphological level (Ilisei et al., 2010; Ilisei and Inkpen, 2011), the following morphological features are proposed:

9. Proportion of nouns over tokens.
10. Proportion of prepositions over tokens.
11. Proportion of pronouns over tokens.
12. Proportion of stopwords over tokens. Stopwords are extracted according to the list of stopwords in the NLTK[35] toolkit.
13. Finite verb rate. This refers to the proportion of finite verbs in texts.
14. Grammatical cohesion rate. This is the proportion of grammatical words over lexical words. Grammatical words are represented by determiners, articles, prepositions, auxiliary verbs, pronouns, conjunctions and interjections. Lexical words are represented by nouns, verbs, adjectives, adverbs and numerals.
15. Individual function words. Each function word in a list is extracted as an individual feature, such as “the”, “of”, “and”, “to”, “be”, “someone”, “self”, etc.
16. Proportion of function words in texts. This is the total number of function words in list normalised by word tokens.

Statistical analysis has always played a major part in the study of plagiarism detection, hence the following statistical features are proposed:

17. Number of sentences.
18. Number of word tokens.
19. Number of characters.

[35] http://nltk.org/

20. Language model 3gram log probability.

21. Language model 3gram perplexity (all tokens).

22. Language model 3gram perplexity (without end of sentence tags).

23. Language model 5gram log probability.

24. Language model 5gram perplexity (all tokens).

25. Language model 5gram perplexity (without end of sentence tags).

And finally, from the linguistic perspective, a syntactic feature that is able to facilitate the investigation of the deeper meaning of text is also proposed:

26. Syntactic tree. Parse trees generated within a document are combined as a whole, and subsequently compared using tree kernels.

To test whether morphological, simplification and statistical features are complementary, the InfoGain attribute evaluator (Formula 4.14 on page 114) is applied and the top 12 features are selected to be tested as a set in the machine learning classification stage, named pre-selected feature set:

- F2: Average sentence length
- F3: Information load
- F6: Proportion of sentences without finite verbs
- F13: Proportion of finite verbs over tokens
- F14: Grammatical cohesion rate
- F19: Number of characters
- F20: Language model 3-gram perplexity (all tokens)

- F21: Language model 3-gram perplexity (without end of sentence tags)
- F22: Language model 3-gram log probability
- F23: Language model 5-gram perplexity (all tokens)
- F24: Language model 5-gram perplexity (without end of sentence tags)
- F25: Language model 5-gram log probability

Once features are selected they are then used as sets of attributes in the machine learning tasks.

7.4 Machine Learning Algorithms

To reiterate the problem of plagiarism direction identification, we proposed two machine learning tasks. The first task is to classify each text (as individual cases) into two classes: plagiarised or original. The second task is to rank a pair of texts (parallel plagiarised and original cases) according to the order in which they were created.

The first task involves two binary classifiers. The classifiers used are as follows: the rule-based learner RIPPER (Formula 4.17 on page 116), and a structured prediction tree kernel Support Vector Machines (SVM).

The symbolic classifier RIPPER was selected as the rules produced by it show which feature contributes most to the learning process. RIPPER [36] was trained and tested with 4-fold cross-validation using various feature combinations that represent simplification, morphological and statistical aspects of texts (see Section

[36] We used the WEKA implementation of RIPPER, the Jrip classifier http://www.cs.waikato.ac.nz/ml/weka/

7.4).

In addition to RIPPER, a structured prediction version of SVM is applied using the SVM-tree kernels (Formula 4.18 on page 117). It is used to perform binary classification of texts according to their syntactic information, using 4-fold cross-validation. The binary classifiers are applied to parallel and non-parallel datasets.

SVMs have been applied to other text classification tasks with success; in particular, the use of an SVM in modelling syntactic information in NLP tasks has aroused interest. Prior to the introduction of tree kernels, syntactic information such as parse trees generated by parsing was difficult to exploit. Ever since Moschitti (2006a,b) developed SVM-Light-TK[37] that allows similarity measurement between two syntactic trees in terms of their sub-trees, it has been applied in tasks such as classifying predicate argument structures as part of semantic role labelling (Moschitti et al., 2006), Question Classification with Semantic Syntactic Tree Kernels (Bloehdorn and Moschitti, 2007), and machine translation (Hardmeier, 2011).

For the second task, SVM-rank (Formula 4.19 on page 117) is used to perform a 4-fold cross-validation on pair-wise ranking between parallel original and plagiarised texts. This ranking metric differs from traditional classification which determines the direction of plagiarism by sorting each pair of texts according to their level of changes. The ranking task is tested with the pre-selected feature set on the parallel datasets, as the ranker cannot be applied to non-parallel cases.

[37] http://disi.unitn.it/moschitti/Tree-Kernel.htm

For the additional multiclass classification task, the RIPPER rule-based classifier is used with 4-fold cross-validation based on pre-selected features.

7.5 Results

The baseline is defined by the proportion of classes from the dataset. As the distribution of classes is 50:50, the proportion of accuracy achieved by chance is 50%.

The rest of the classifications are tested in the following conditions:

1. Rule-based with the pre-selected features.
2. Rule-based with only the simplification features.
3. Rule-based with only the morphological features.
4. Rule-based with only the statistical features.
5. SVM-tree kernels with only the syntactical feature.
6. SVM-rank using the pre-selected features.

Table 7.2 shows the accuracy achieved using various machine learning algorithms. Table 7.3 presents the tree kernels experiment tested with tree and selected features. Table 7.4 gives the detailed tree kernels results. Table 7.5 shows the accuracy achieved using rule-based algorithms based on various feature sets. Table 7.6 gives a breakdown of the precision, recall and f-score of the two classes (original and plagiarised) using various feature sets.

The results are compared amongst the types of classification. It is observed that both rule-based classification (RIPPER) and pair-wise linear ranking (SVM-

Metric	**Simulated**	**Parallel Artificial**	**Non-parallel Artificial**
Baseline	50%	50%	50%
Rule-based Pre-selected features	**75.66%**	**97.94%**	98.38%
SVM-tree kernels Syntactic feature	56.17%	79.9%	**99.45%**
SVM-rank Pre-selected features	74%	95%	-

Table 7.2: Comparison of the accuracy in classification and ranking tasks

rank) using selected features performed well, although rule-based classification is slightly better. However, the performance of syntactic tree kernels (SVM-tree kernels) did not perform as well as the other two, but the result is still above chance-level. Moreover, tree kernels on the parallel artificial dataset outperformed the simulated dataset by over 20% as artificial cases exhibit less linguistically well-constructed texts, which correspond to the findings of Grozea and Popescu (2010), and also confirms the hypothesis that artificial plagiarism cases display significant and measurable differences in relation to their original.

Feature	**Precision**	**Recall**	**Accuracy**
Tree only	56.3% ± 1.3%	55.5% ± 2.6%	56.2% ± 0.9%
Tree and vector	56.3% ± 0.5%	57.7% ± 4.1%	56.4% ± 0.6%

Table 7.3: Syntactic tree kernels tested with tree only and tree plus selected features

Table 7.3 shows the results obtained from the simulated dataset. The syntactic tree kernels combination of tree plus feature vectors did not bring significant improvement over using tree alone. The vectors tested included average sentence length, information load, functional words over lexical words and proportion of finite verbs over tokens. Hence, further experiments on tree kernels were tested with tree only (Table 7.4).

Dataset	**Precision**	**Recall**	**Accuracy**
Simulated	56.3% ± 1.3%	55.5% ± 2.6%	56.2% ± 0.9%
Parallel Artificial	83.7% ± 0.8%	74.3% ± 1.2%	79.9% ± 0.9%
Non-parallel Artificial	99.9% ± 0.1%	98.9% ± 0.2%	99.5% ± 0.1%

Table 7.4: Syntactic tree kernels tested on tree only across dataset

The tree kernels performed particularly well in improving the precision of parallel artificial plagiarism texts, which indicates that it is suited to correctly identifying the original cases, thereby reducing the false positive cases. This again shows that artificial plagiarism cases display significant differences from original and manually simulated texts.

Feature set	**Simulated**	**Parallel Artificial**	**Non-parallel Artificial**
All features	74.67%	**98.15%**	**99.5%**
Pre-selected features	**75.66%**	97.94%	98.38%
Simplification features	59.81%	70.24%	71.6%
Morphological features	59.53%	68.08%	97.38%
Statistical features	74.17%	97.78%	98.41%

Table 7.5: Accuracy of various feature sets classified by the rule-based classifier

The comparative results of different feature sets show that it is a much easier task to classify plagiarised texts from original texts when the cases are not parallel. This is due to the significant statistical and morphological differences between the texts. Unsurprisingly, the syntactic feature classified by SVM-tree kernels performed particularly well, outperforming other feature sets with the exception of all features.

The simplification features on the non-parallel dataset did not perform as well as they did in the parallel datasets. This shows that simplification-based features do have an impact on the identification to plagiarism direction, although the impact may be too insignificant and trivial in comparison which more effective features such as syntactic tree-kernels.

To take a closer look at the performance of various feature sets, the detailed breakdown of precision, recall and F-score is listed in Table 7.6.

The results of the simulated dataset did not show significant differences between the original and plagiarism classes using various features. In the parallel artificial dataset some levels of differences are observed when using simplification-based and morphological features. Differences are also observed in the non-parallel artificial dataset using simplification features. There is a trend that the simplification features improve the recall in detecting artificial original texts, and improve the precision in detecting artificial plagiarised texts.

The results of the non-parallel artificial dataset largely agree with the parallel artificial dataset, with the exception that morphological features perform significantly better in this experiment. In the parallel artificial dataset, morphological features achieved an accuracy of 68.08% while the morphological features of the non-parallel artificial dataset achieved over 91% accuracy. This is due to the characteristics of the non-parallel original and plagiarised texts being directly reflected in the morphological features.

In the comparison between the types of feature, it is observed that using sta-

Dataset	**Class**	**Feature set**	**Precision**	**Recall**	**F-score**
Simulated	Original	Pre-selected	**75.80%**	75.40%	**75.60%**
		Statistical	73.60%	**75.50%**	74.50%
		Simplification-based	59.90%	59.40%	59.70%
		Morphological	59.80%	58.20%	59.00%
	Plagiarised	Pre-selected	**75.50%**	**75.90%**	**75.70%**
		Statistical	74.80%	72.90%	73.80%
		Simplification-based	59.70%	60.20%	60%
		Morphological	59.30%	60.80%	60%
Parallel	Original	Pre-selected	**98.40%**	97.50%	**97.90%**
		Statistical	97.80%	97.70%	97.80%
		Simplification-based	67.80%	72.20%	72.20%
		Morphological	66.10%	74.10%	69.90%
Artificial	Plagiarised	Pre-selected	97.50%	**98.40%**	**97.90%**
		Statistical	**97.70%**	97.80%	97.80%
		Simplification-based	73.50%	63.30%	68%
		Morphological	70.50%	62.10%	66%
Non-parallel	Original	Pre-selected	98.5%	**98.3%**	98.4%
		Statistical	**98.6%**	98.2%	98.4%
		Simplification-based	69.7%	76.4%	72.9%
		Morphological	97.1%	97.7%	97.4%
Artificial	Plagiarised	Pre-selected	**98.3%**	98.5%	98.4%
		Statistical	98.2%	**98.6%**	98.4%
		Simplification-based	73.9%	66.8%	70.2%
		Morphological	97.7%	97.1%	97.4%

Table 7.6: Precision, recall and F-score of various feature sets in two classes using rule-based classifier

tistical features alone yield very high performance, which is only slightly less than the best features. The attribute evaluator shows that the features involving statistical language models are most contributing, with the exception of number of characters and word tokens in the manual dataset which ranked higher than some language model features. This may be due to the corpus design - as most original segments are longer than their plagiarised counterparts, the length of text may already be a good indicator.

The statistical features on all datasets outperformed simplification and morphological features, which shows that statistical features are suitable in either manual or artificial cases. Furthermore, statistical features on the artificial dataset performed around 27% higher than simplification and morphological features, while it performed around 14% higher on the manual dataset. This suggests that statistical features on the manual dataset play a lesser role in detection in comparison to that of the artificial dataset, though the improvement is still significant.

To determine the optimal amount of training and testing data for the experiment, the learning curves for simulated, parallel artificial and non-parallel artificial datasets are shown in Figure 7.2.

Even when the size of the training and testing data are reduced drastically, the accuracies of the artificial datasets are still very high. This suggests that the differences between original and artificial texts are very significant and should not require extensive training data for a model to be built. As shown in the figure, even with only 250 examples for training and testing, the accuracies for both artificial datasets already reached over 90%. For the simulated dataset, the more examples the better the accuracy is.

For the additional multiclass classification task, the pre-selected feature set was applied with the rule-based classifier in a 4-fold cross validation. The accuracy

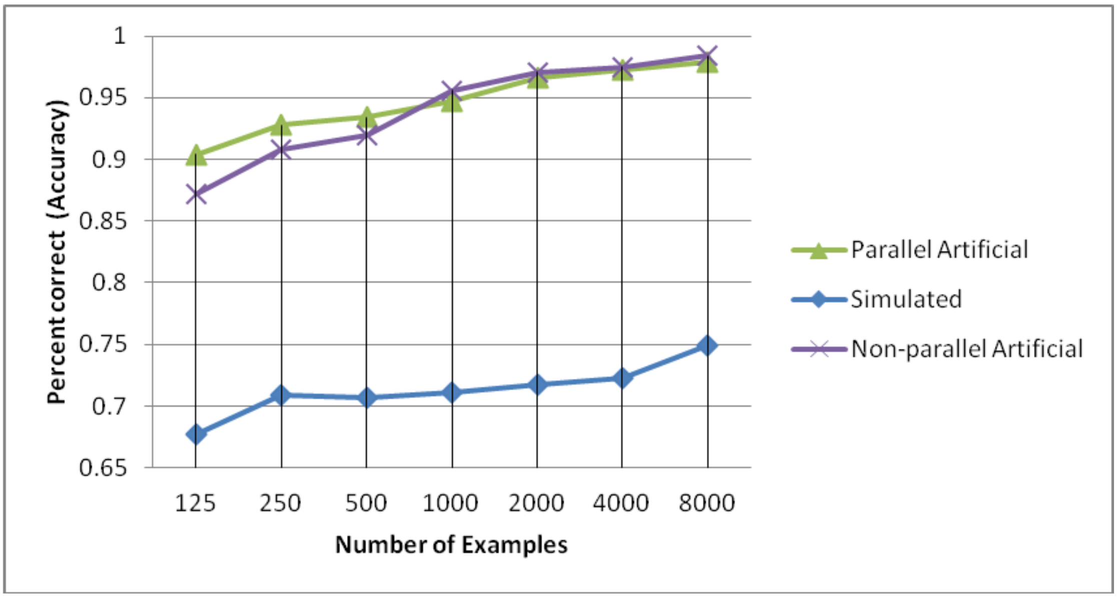

Figure 7.2: Learning curve showing the accuracy with various sample sizes

reached 85% in the three-class classification task. Table 7.7 shows the breakdown of the scores in each class.

Class	Precision	Recall	F-score
Simulated	74.80%	68.90%	71.70%
Artificial	96.90%	95.10%	95.80%
Original	83.90%	87.90%	85.90%

Table 7.7: Precision, recall and F-score across classes

The results show that it is a much simpler task to classify simulated texts when both artificial texts and original texts are present. This is in agreement with the results on testing the number of examples needed to improve the classification for simulated cases - the more examples the better it will be. Figure 7.3 shows the learning curve for the multiclass classification task.

The best sample size may be around 4,000 cases as the learning curve is shal-

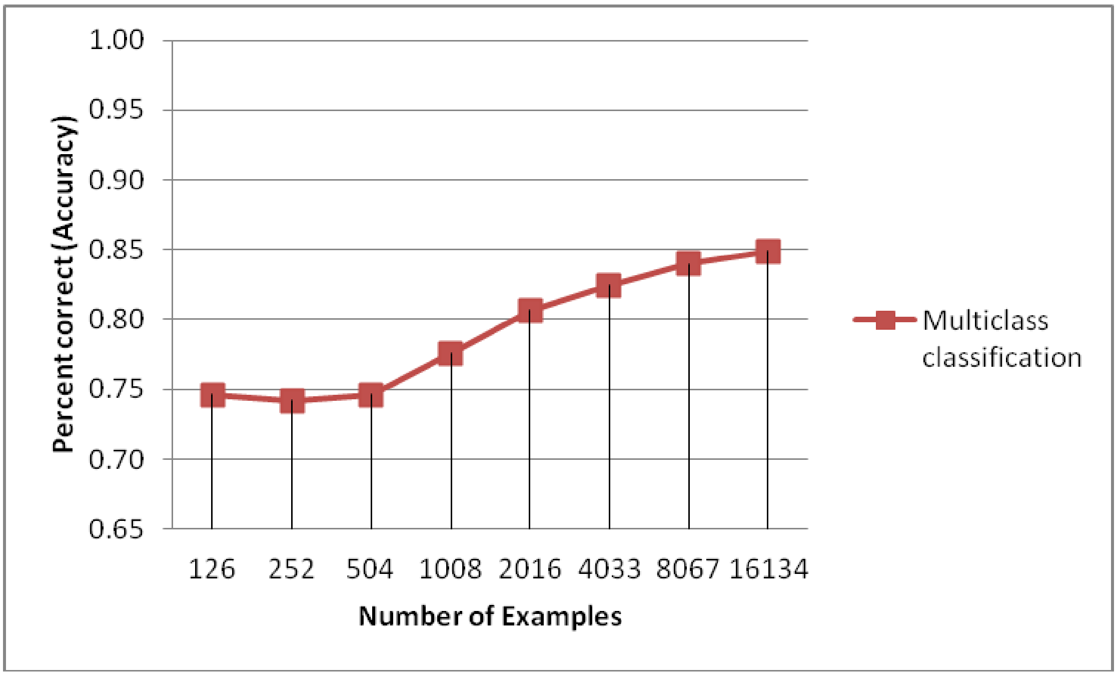

Figure 7.3: Learning curve showing the accuracy for multiclass classification with various sample sizes

lower when more examples were added. The results indicate that the addition of other types of texts can help to improve the classification of simulated texts, as more distinctive traits from artificial texts can highlight their differences.

A closer inspection of the pre-selected features is presented in order to evaluate their individual effectiveness. The top 12 features identified by the InfoGain attribute evaluator from both the simulated and parallel artificial datasets were selected as a set. These features represent the morphological, statistical and simplification traits, and their respective InfoGain scores are listed in Figure 7.4.

Statistical features are very effective indicators in classifying both manually simulated and artificially generated plagiarised cases from original texts. Morphological and simplification features did not rank as highly, but still contributed to

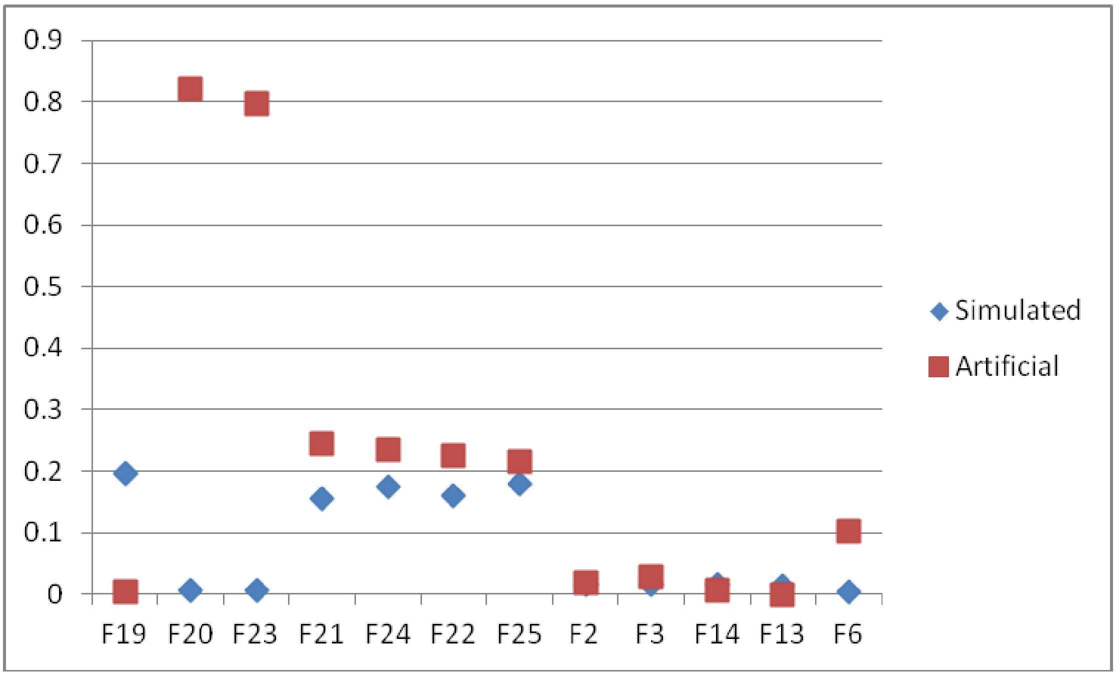

Figure 7.4: InfoGain scores for the pre-selected features in the datasets

the classification. The list of top features ranked by InfoGain for the datasets can be found in Appendix D Table D.2.

In particular, language model 3-gram and 5-gram log probabilities performed significantly better in the parallel artificial set. On the other hand, the language model 3-gram and 5-gram perplexity are ranked as the best features for the simulated set. The 3-gram and 5-gram perplexity are fairly consistent across all datasets. However, the assumption that 3-gram works better on shorter texts and 5-gram works better on longer texts does not hold up. In the datasets, simulated cases are shorter and artificial cases are longer, but 5-gram LM features turned out to rank higher than 3-gram features in the simulated set, and vice versa on the artificial set. Further tests on longer simulated texts and shorter artificial texts are suggested in order to fully investigate this matter.

The high performance of the language model features raises the question of whether the domain of the training data should be considered. The language model was trained using the original texts from the PAN-PC-10 corpus, which contains full texts of books from project Gutenberg[38]. Hence, the domain of the corpus is English literature. The question is this: if the language model in this experiment is not trained with texts from English literature, will that make a significant difference? The hypothesis is that in a realistic scenario, not all original documents would be available or belong to the same domain, and the language model will have to be trained with an out-of-domain dataset. To investigate this, an additional experiment using a domain-independent dataset to generate the statistical features using language models is performed. The dataset is composed of the European Parliament parallel corpus[39], and the English version with 1.5 million sentences was used as the training model. The size of this corpus is similar to that of the in-domain corpus with PAN-PC-10 original texts, which has 1.7 million sentences. Table 7.8 and Table 7.9 show the results tested on the simulated dataset and the parallel artificial dataset, respectively.

The results trained with the out-of-domain language models show a drop in accuracy on the simulated dataset, but for the artificial dataset the difference is minor. The out-of-domain statistical features helped to reduce the number of false negatives in the simulated dataset original class, but increased the number of false

[38] http://www.gutenberg.org/

[39] http://www.statmt.org/europarl//

Domain	Dataset	Class	Precision	Recall	F-score
In-domain	Simulated	Original	**73.60%**	75.50%	**74.50%**
		Plagiarised	**74.80%**	**72.90%**	**73.80%**
	Artificial	Original	97.80%	**97.70%**	**97.80%**
		Plagiarised	**97.70%**	97.80%	**97.80%**
Out-of-domain	Simulated	Original	65.40%	**77.20%**	70.80%
		Plagiarised	72.20%	59.20%	65.10%
	Artificial	Original	**98.10%**	96.00%	97.00%
		Plagiarised	96.10%	**98.10%**	97.10%

Table 7.8: Comparison of results of the statistical features using different language models

Domain	Dataset	Accuracy
In-domain	Simulated	**74.17%**
	Artificial	**97.78%**
Out-of-domain	Simulated	68.21%
	Artificial	97.04%

Table 7.9: Comparison of accuracies of the statistical features using different language models

negatives in the plagiarised class. The experiment shows that morphological and simplification features are more robust that domain-dependent statistical features in identifying simulated cases. For identifying artificial cases, a corpus of any domain is equally effective. A selected group of empirical examples from the datasets are shown below:

Example 1: Correctly classified pair from simulated dataset

Source: “But a better idea of the journal can perhaps be given, by stating what it lacked than what it then contained. It had no leaders, no parliamentary reports, and very little indeed, in any shape, that could be termed political news.”

Plagiarised: "The journal could better be described by what was missing than what it contained. It lacked leaders, had no parliamentary reports and in no way could be described as political news."

Example 2: Correctly classified pair from artificial dataset

Source: "A dispatch from the Headquarters Staff of the Commander in Chief says: At the beginning of March, (Old Style,) in the principal chain of the Carpathians, we only held the region of the Dukla Pass, where our lines formed an exterior angle."

Plagiarised: "A chain of the Carpathians, we only held the region from the Commander in Chief says: Of the Dukla Pass, where our lines lived didn an space of March, (Old Property,) in the dispatch at the commencement of the Cause."

Examples 1 and 2 demonstrate simplification and morpho-syntactic traits, which include joining and splitting sentences, and synonym substitution. These clues are sufficient for the algorithm to determine the direction of plagiarism.

Example 3: Incorrectly classified pair from simulated dataset

Source: "There is a great gain in time of acceleration and for stopping, and for the Boston terminal it was estimated that with electricity 50 per cent, more traffic could be handled, as the headway could be reduced from three to two minutes."

Plagiarised: "There is a huge profit in time of speeding up and for slowing down, and for the Boston extremity it was guessed that with current 50 percent, more movement could be lifted, as the headway could be minimised from three to two minutes."

Example 4: Incorrectly classified pair from artificial dataset

Source: "'Giulietta,' at last said the young man, earnestly, when he found her accidentally standing alone by the parapet, 'I must be going to-morrow.' 'Well, what is that to me?' said Giulietta, looking wickedly from under her eyelashes."

Plagiarised: "'well, what is that to me?' said Giulietta, standing alone under the parapet, earnestly, when he found her were accidentally looking wickedly from by her eyelashes. 'Giulietta,' at last young the man, 'it must be going to-morrow.'"

Example 3 does not contain any simplification traits but only synonym substitution. Example 4 involves sentence swapping without any word-level changes. These two examples are misclassified as the algorithm cannot distinguish between the original and the plagiarised segments without sufficient linguistic clues.

7.6 Discussion

As manually simulated cases are not created as translated texts per se, it is questionable whether translationese and the Translation Universals are applicable. However, from the experimental results and examples shown above, one can observe a trend that some level of improvement is gained, by using a combination of simplification, morphological and statistical features. The results provide support that the features are an effective framework, but there is no concrete evidence that the Translation Universals or translationese fit the plagiarism direction detection scenario perfectly.

This study shows that there are indeed traits of plagiarism that are present in texts. Whether these traits concur with translationese and the Translation Universals or not, they do help to distinguish between original and plagiarised texts. This finding should establish the foundation for future developments in intrinsic plagiarism detection, authorship attribution and cross-lingual plagiarism detection. The principle of intrinsic plagiarism detection is to identify segments of texts from a document without the references from original documents. By treating each segment of the text as an individual instance, the proposed methodology in this thesis may be able to identify patterns to distinguish between non-plagiarised and plagiarised text within a document. Similarly, for authorship attribution tasks, the writing traits of individual authors can be collated into various patterns, thereby establishing learning models for identifying texts that may fit specific writing styles.

The traits are represented by simplification, morphological, syntactic and statistical features. The features were investigated with a supervised machine learning approach, which was employed to distinguish between original and plagiarised texts. The results showed that original and rewritten texts exhibit distinguishable traits, which can be characterised by statistical and linguistic features and measurable via computational means. An analysis of the features was performed on a manually simulated plagiarism dataset, a parallel artificially generated plagiarism dataset and a non-parallel artificial dataset. The accuracies of the selected feature set that includes a combination of simplification, morphological and statistical features on the simulated and parallel artificial datasets were 75.66% and 97.94% respectively, which is very satisfactory and well above chance-level.

The results also showed that statistical features alone can reach a high accuracy, and in particular, the features involving the use of language models are very effective in both datasets. Training the language model with an in-domain dataset yielded better performance for the simulated dataset, but there was very little difference for the artificial dataset. The syntactic feature used in tree kernels showed significant improvement when applied to the artificial datasets, which confirms the hypothesis that artificial cases are less syntactically well-constructed. In particular, the results from the parallel artificial dataset showed a significant improvement by using tree kernels to improve precision. This is due to the syntactically well-formed original texts displaying significant differences to the artificially plagiarised texts, thereby increasing the chance for original texts to be correctly identified and thus reducing the number of false positive cases. For the learning approach based on pair-wise comparison, the accuracies of the simulated and parallel artificial datasets are 74% and 95% respectively, which again are satisfactory, although its performance is slightly less than the binary classification approach based on individual instances.

It is expected to see artificial cases displaying significant differences over simulated cases in all areas. Although previous studies used statistical means, this study incorporated linguistically motivated features together with statistical means. Unsurprisingly, artificial texts exhibit less linguistically coherent texts and therefore features that are based on syntactical tree kernels performed much better than on manual cases. Along with statistical features, it is possible to identify not only the direction of plagiarism, but also the type of plagiarism, as shown in the multiclass classification experiment.

The differences between simulated, artificial and original texts are emphasised in the multiclass classification task, where the pre-selected features effectively distinguished between the three classes of texts with an accuracy of 85%. Overall, the study confirms the hypothesis that original texts and plagiarised texts exhibit significant differences which are measurable via computational means.

Finally, as Translation Universals may better fit with cross-lingual plagiarised texts than monolingual plagiarised texts, applying translationese features in cross-lingual plagiarism detection is certainly an interesting direction for future studies. Current studies mainly use machine translation to translate both plagiarised and original texts into one target language in order to facilitate similarity comparison. Following related work in translation studies, language models that are compiled from the original and translated texts can also be utilised in plagiarism detection, along with translationese traits that can help to identify translated segments within texts.

7.7 Summary

This chapter presented our proposed framework in the under-explored research area of detecting plagiarism direction, which met the **fourth objective**. The aim was to distinguish original texts from rewritten texts, with application to plagiarism detection. Different from traditional plagiarism detection tasks, the proposed framework does not involve the standard approach of conducting exhaustive comparisons between all suspicious and source texts. It instead focuses on the sub-problem of finding segments that exhibit rewriting traits. In addition, the framework investigates traits that are inspired by studies on translationese, Translation Universals and translation direction detection.

The findings of this study can be directly used to improve the performance and reduce the computational cost of the filtering stage, and the resources can instead be focused on the more complex comparisons in subsequent stages. This study can also benefit other related fields such as cross-lingual plagiarism detection, as it can highlight potentially plagiarised segments that have been translated. In addition, it can be applied in intrinsic plagiarism detection or authorship attribution tasks, due to its ability to detect segments of text that are incoherent with the rest of the text within a document. Furthermore, this study lays the foundation for further research on text reuse, as the SVM-rank algorithm can be extended to cover multiple versions derived from the same original text.

Chapter 8

Conclusions

This chapter summarises this study and provides an outline of further research directions. Section 8.1 presents a summary of the main research findings of preceding chapters, and reviews the main contributions of this study. This chapter concludes with Section 8.2, which provides an insight into how the contributions in this thesis could be applied to the continuous development of plagiarism detection and other related fields.

8.1 Review of the Contributions

To recapitulate, the aim of this study was defined as four research questions:

- How can Natural Language Processing techniques be incorporated into existing approaches?
- Does machine learning bring any benefits to the plagiarism detection framework?
- Will the framework perform well in a small-scale scenario as well as a large-scale scenario?
- Can the task of identifying the direction of plagiarism benefit from the investigation of statistical and linguistic traits?

To answer the research questions, four main objectives were set. The thesis was organised in two main parts, where part 1 (Chapters 2 and 3) provided the background of the thesis by defining the terminologies used in plagiarism studies, and providing a comprehensive review of existing approaches. Part 2 (Chapters 4-7) described the proposed framework and experiments which corresponded with the objectives.

Chapter 1 introduced the plagiarism challenge which motivated this study and defined the scope, aims and objectives. It provided an introduction to the proposed framework and an overview of the thesis. Chapter 2 defined the important concepts of plagiarism in the research context as well as the terminologies used in existing studies, which were also used in this thesis. The chapter briefly outlined various types and characteristics of plagiarism, and concluded with general evaluation approaches.

The first question was answered by a comprehensive review of the existing plagiarism detection approaches in Chapter 3 and the proposal of an NLP-based plagiarism detection framework in Chapter 4. Chapter 3 reviewed the limitations of the existing approaches and described the role of NLP in plagiarism detection. Other related work that provided inspiration for the proposed framework was also reviewed.

Chapter 4 described our proposed framework for external plagiarism detection. The framework used techniques identified in related studies to produce a robust solution to the plagiarism detection tasks. The five-stage plagiarism detection approach included various combinations of NLP techniques, similarity metrics and machine learning algorithms. The chapter concluded with a list of the conventional evaluation metrics which were used for analysis in subsequent chapters. This fulfilled the **first objective**, which was the incorporation of shallow and deep NLP techniques into a plagiarism detection framework.

The thesis answered the second and third questions in the initial experiment (Chapter 5) and subsequent experiment (Chapter 6) where NLP techniques and machine learning algorithms were successfully applied to improve n-gram based plagiarism detection approaches.

Chapter 5 fulfilled the purpose of an initial study and identified the most beneficial techniques for further experiments. The experiment was performed on a small-scale corpus, and the results suggested that for short, slightly-modified texts, it is not necessary to apply deep techniques, as string-matching algorithms are sufficient to achieve a satisfactory result effectively. This met the **second objective**, which evaluated the proposed framework with a machine learning model. The overlapping n-gram with text pre-processing and shallow NLP techniques were able to distinguish between clean and plagiarised documents with ease. However, deeper techniques may be more useful in helping to distinguish between different levels of plagiarism, and plagiarised cases with substantial changes. Some of the features tested can be seen as a framework for language-independent detection. One of the most successful features was based on dependency parsing. Parsers for various languages could be explored for cross-lingual plagiarism detection tasks.

The third research question was answered by the experiments in Chapter 6, where shallow and deep NLP techniques were applied to corpora containing various case lengths. This fulfilled the **third objective** to evaluate the scalability of the proposed framework in various experimental settings. They followed the initial experiment (Chapter 5) and further explored other NLP techniques. An in-depth analysis was performed at the document level and at the passage level, using a corpus with clear and distinctive plagiarism classes. The experiments confirmed the hypothesis that deep NLP techniques can improve the identification of plagiarised text, as the techniques helped to reduce the false negative cases. It was discovered that the effectiveness of deep techniques correlated with case length. This suggested that the best application of string-matching and shallow processing techniques is at the document level, and that deep processing techniques should be applied at the passage level. The discovery is also related to the next research question of how NLP techniques can influence the detection of plagiarism direction and the filtering of candidate documents.

The final research question was answered in Chapter 7, which described the innovative experiment on identification of plagiarism direction performed on original and rewritten text passages. The proposed framework integrated linguistic and statistical traits with machine learning algorithms. Instead of following a traditional brute-force pair-wise comparison approach, the experiment focused on fitting individual texts into their respective class patterns. The results showed that the identification of plagiarism direction can be easily performed using statistical and linguistic features. These features showed promising results even when they were tested on manually rewritten texts that are challenging for human beings to identify. In particular, the statistical features involving the use of language models can reach a high accuracy. The syntactic feature used in SVM-tree kernels also delivered significant results. This fulfilled the fourth and **final objective**, which was to propose and evaluate a framework for identification of plagiarism direction.

On the challenge of filtering candidate documents, one of the main issues of the plagiarism detection approach is that the mechanical means cannot prove the absence of plagiarism. Instead, the approach can only provide indications as to what parts of the text might have been copied from a potential source. This also comes with a large quantity of false positives. From our experiments, it is obvious that the techniques used in the filtering stage must be efficient and effective in reducing the number of false negatives, but not at the expense of increasing the false positives. The results from Chapter 6 showed that string-matching and shallow techniques could be good indicators as a filtering approach, but that the cut-off threshold must be set appropriately in order to maintain a good balance between precision and recall. This issue is also related to the experiment in Chapter 7. Unlike the conventional plagiarism detection approach where one text must be compared with all texts, the proposed direction detection framework based on building patterns from linguistic and statistical traits can be utilised as an enhanced filtering approach. Even though the intention of this research is not the creation of a plagiarism detection system that eliminates the necessity of human intervention altogether, the promising results in this study have shown that it is possible to reduce such a need.

In summary, the objectives of this study have been met and are listed as the following contributions:

1. The proposal of a novel plagiarism detection framework that incorporates string-matching approaches with NLP techniques. This framework not only employed shallow comparison of texts such as overlapping n-grams, but also investigated deeper linguistic features such as syntactic structure and semantics using deep NLP techniques.
2. The exploration of the role of machine learning approaches in plagiarism detection. The evaluation from the empirical studies showed that machine learning is an essential part of the framework.
3. The application of the proposed framework in a small-scale scenario and a large-scale scenario, with experiments performed at the document level and at the passage level. This evaluated the scalability of the framework and also identified the best techniques for varied case lengths.
4. The integration of statistical and linguistic features in the identification of plagiarism direction. The proposed framework provided a novel perspective on plagiarism detection, where individual plagiarism cases were characterised by patterns built from linguistic and statistical traits, and the process no longer relies on brute-force, pair-wise comparison.

A final note is that even with the routine use of plagiarism detection systems, using one system alone is not enough (Evans, 2006). Plagiarism detection on a large scale is very difficult to sustain as the number of positive cases will require more human resources to investigate them. Therefore, a plagiarism detection system should minimise the amount of cases mistakenly marked as plagiarism.

A plagiarism detection system should be able to identify all possible plagiarised cases for further manual investigation, as it is impractical to rely fully on detection systems to determine academic integrity. Ultimately, a plagiarism detection system can only suggest what has been plagiarised, but cannot give a final verdict.

8.2 Further Work

The preceding sections described the current state of research presented in this thesis. The study may provide inspiration for future research directions and potential extensions. The two main directions that are described in this section are: i) cross-lingual plagiarism detection, and ii) ranking multiple versions of plagiarised texts.

8.2.1 Cross-lingual plagiarism detection

The study described in this thesis is focused on monolingual English text segments. This section provides an insight into further studies on cross-lingual plagiarism detection, based on the framework established in Chapter 4.

Cross-lingual plagiarism detection has started to receive attention in recent years. Existing approaches rely on generalising texts into one language for further processing. This normally involves the use of machine translation tools. For example, in Muhr et al. (2010), the first step of pre-processing is to determine whether the original texts are in English. If not, the next step is to determine the language of the original texts, and then translate the original texts into English. Comparisons between original texts and suspicious texts can then be performed.

Another example is the PAN-PC-10 corpus which contains English, Spanish and German original documents, with all plagiarism cases from Spanish and German documents either mechanically or manually translated into English. The method adopted by PAN competitors to detect plagiarism from non-English documents was to use machine translation tools to translate Spanish and German source documents into English. For example, machine translation tools such as TextCat were used in the PAN competitions in the pre-processing stage (Kasprzak and Brandejs, 2010). This approach means that detection accuracy is in turn limited by the performance of the language identification tool and machine translation tool.

Understandably, machine translated texts are not 100% grammatically correct and it would not be desirable to apply deep linguistic analysis on such texts. However, as the use of human translators is infeasible in large-scale detection scenarios, the use of machine translation tools is still the standard approach in existing studies. The challenge is that in order to implement NLP techniques in cross-lingual plagiarised texts, the translated texts must be able to be interpreted by NLP techniques. The texts must be syntactically correct and must not contain any foreign characters.

Another existing approach of cross-lingual natural language processing is to use statistical alignment with a bilingual thesaurus. Pinto et al. (2009) described their use of IBM M1 alignment model for such a task. Similarily, (Potthast et al., 2010a) suggest using statistical alignment in cross-lingual plagiarism detection. The languages in the document collection include English, German, Spanish, French, Dutch and Polish, and the translation was performed with a statistical bilingual dictionary and aligned using the IBM M1 alignment model. As the model was successfully applied in other monolingual and cross-lingual information retrieval tasks, the authors implemented the sentence alignment model in cross-lingual plagiarism detection, and performed experiments on a parallel corpus. The keywords were extracted from the document collection in the information retrieval process. To measure the similarity, information retrieval models were applied in the detailed analysis stage, using three retrieval models that included character 3-grams, semantic analysis and alignment-based analysis. Their experiment showed that information retrieval can be used as a basic strategy for cross-lingual plagiarism detection. However, as the corpus used was constructed from the European Parliament parallel corpus and Wikipedia, it is not certain whether real plagiarism cases would be as easy to detect per se, as cross-lingual plagiarism cases are often not aligned as a parallel corpus.

Another example of cross-lingual plagiarism detection by Ceska et al. (2008) was also based on the European Parliament parallel corpus. The approach was based on analysis of word positions and words were translated using EuroWordNet. The techniques used include semantic-based word normalisation, and the generalisation of synonyms before indexing. The experiment showed that the method using tf-idf was outperformed by the method using Singular Value Decomposition (SVD), a technique derived from Latent Semantic Analysis (LSA).

These studies used parallel corpora in their experiments. Although alignment with a bilingual dictionary was an effective approach, it is only so because parallel corpora provide near-duplicate sentences that are relatively easy to align. The task was made easy as the keywords can be translated word-for-word via machine translation. The challenge is that if the texts were to be paraphrased, word alignment and machine translation would not be easy tasks. The difficulty in obtaining a cross-lingual plagiarism corpus poses another challenge.

To bring a new perspective to cross-lingual plagiarism detection, we refer to our framework of plagiarism direction detection described in Chapter 7. In the framework, we proposed the incorporation of translationese and Translation Universals in identifying individual text cases, which does not require pair-wise comparison between original and suspicious texts. The framework can be expanded for cross-lingual plagiarism detection as it does not require knowledge of the original language. Instead, based on translationese and Translation Universals, characteristics of a language can help to distinguish translated texts from non-translated texts. In other words, the detection framework does not rely on the traditional approach of identifying and translating original languages from the original texts collection. Detection can be performed based on the linguistic features of the suspicious texts alone, by identifying traits of translation and then suggesting the original language of the suspicious texts. This can be an effective filtering strategy to narrow down the search span for further similarity comparison.

8.2.2 Versions of plagiarism

This section provides an insight into further investigations based on the findings of Chapter 7. The detection of plagiarism versions can be explored in a number of ways. These include analysing the impact of the text domain on the language model features, experimenting with other types of rewritten texts with more than one version for each original text, and different levels of text reuse, similar to the Measuring Text Reuse (METER) experiments in the journalism domain by Gaizauskas et al. (2001).

As mentioned in Section 3.4, the METER corpus was created for measuring text reuse in the journalism domain. The work by Clough et al. (2002b) uses three approaches of supervised machine learning to distinguish original from derived newswire texts. It is important to note that the principles of measuring text reuse and plagiarism detection are different. In measuring text reuse, the source of the corpus was from the Press Association, which represents the original source data. The re-worded news by news agents represents the rewritten data. As it is not the intention of journalists to "plagiarise" pieces of news, they do not try to conceal the fact that the texts are not original. Therefore, the nature of journalism text reuse and plagiarism detection is different and should be considered when designing experiments.

The proposed methodology in Section 7.1 can be applied in the identification of text versions. It is capable of detecting whether a text is original or in a modified form, and of determining the original from a pair of texts using the SVM-rank algorithm with statistical and linguistic features. In particular, the framework successfully classified between the manually simulated, artificially generated and original texts classes in the multiclass classification with an accuracy of 85%. This could be an effective filtering strategy for realistic plagiarism detection systems, as once a document is determined original there is no need to perform further processing.

8.2.3 Plagiarism meets paraphrasing

As the most challenging plagiarism cases normally contain paraphrases, Barrón-Cedeño (2012); Barrón-Cedeño et al. (2013) gave an insight into detecting paraphrases in plagiarism cases, using a subset of the PAN-PC-10 corpus.

The P4P corpus[40] consists of the short simulated plagiarism cases in the PAN-PC-10 corpus. It contains 847 pairs of sentences of 50 words or less, with 20 types of paraphrase manually annotated. The annotation is based on linguistic units: words, phrases, clauses, and sentences. The paraphrase types include morphological, lexical, syntactic, discourse, semantic changes and other changes. Of these paraphrase types, the most frequent type is lexical changes, such as changes in spelling and format, word substitutions with synonyms and antonyms, and compounding, adding and deleting words.

The aim of these studies is to investigate which types of paraphrases pose more challenge to the existing plagiarism detection systems in PAN'10. The analysis was based on those systems and it was discovered that system performance drops when it comes to complex paraphrases. Word substitution with the same polarity, such as synonym replacement, general/specific substitutions or exact/approximate alternations, and additions/deletions are the most common paraphrase types. Furthermore, plagiarised text fragments normally are shorter than their

[40] http://clic.ub.edu/corpus/en/paraphrases-en#

source texts. These findings are also observed in our experiments, where linguistic analysis such as lexical generalisation using WordNet helps to identify substituted words, and statistical features based on the length of the texts helps to distinguish between plagiarised and original texts.

Currently, our proposed framework does not perform segmentation, i.e., the framework does not pinpoint the exact location of the plagiarised texts within a case, but instead it classifies each case into its respective class. Segmentation is outside the scope of this study as the aim is to investigate the impact of linguistic processing, and segmentation poses a different problem with its own challenges. We chose to investigate plagiarism detection as a classification task, and it was stated at the beginning of the thesis that the goal is to distinguish between original and rewritten texts. Besides, the application of deep NLP techniques does not always preserve the word order, hence it is difficult to cross-reference the similarity matches between various features with their original positions in the document.

One possible direction for future work would be to identify the location of the plagiarised texts, along with deep linguistic analysis that may improve existing plagiarism detection methods. The proposed framework could be revised to incorporate techniques which can maintain word order as well as representing multiple features. This will allow analysis to be based on the segment level within a case and identify plagiarised texts in a fine-grained approach.

This thesis addressed the aims and objectives which were motivated by the intention to resolve the plagiarism detection challenge. The goals and the research questions were answered in the preceding chapters and the four main contributions of the study were presented. The proposed framework for NLP in plagiarism detection and plagiarism direction were designed based on substantial literature reviews. The implementation and comparative evaluation fitted the purpose of an investigative study and the topics for further development based on the research findings were explored. The final remark is that even though plagiarism detection tools will help to make a detection task easier, they cannot give the final judgement on determining whether a document is plagiarised or not. The final decision in plagiarism detection should therefore always involve human judgement.

Bibliography

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity. In Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM), pages 385–393, Montreal, Canada, 2012. ACL.

Salha Alzahrani and Naomie Salim. Fuzzy Semantic-Based String Similarity for Lab Report for PAN at CLEF 2010. In Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2010), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'10), Padua, Italy, 2010.

Einat Amitay, Sivan Yogev, and Elad Yom-Tov. Serial Sharers: Detecting Split Identities of Web Authors. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 11–18, Amsterdam, Netherlands, 2007. ACM.

Ion Androutsopoulos and Prodromos Malakasiotis. A Survey of Paraphrasing and Textual Entailment Methods. Journal of Artificial Intelligence Research, 38(1): 135–187, 2009.

William H. Angoff. The Development of Statistical Indices for Detecting Cheaters. Journal of American Statistical Association, 69(345):44–49, 1974.

Jo Badge and Jon Scott. Dealing with plagiarism in the digital age. Technical report, School of Biological Sciences, University of Leicester, Leicester, UK, 2009.

Mona Baker. Corpus Linguistics and Translation Studies: Implications and Applications. Text and Technology: In Honour of John Sinclair, 1993.

Mona Baker. Linguistics and Cultural Studies: Complementary or Competing Paradigms in Translation Studies? Übersetzungswissenschaft im Umbruch: Festschrift für Wolfram Wilss, pages 9–19, 1996.

Jun-Peng Bao, Jun-Yi Shen, Xiao-Dong Liu, Hai-Yan Liu, and Xiao-Di Zhang. Finding Plagiarism Based on Common Semantic Sequence Model. In Proceedings of the International Conference on Web-Age Information Management (WAIM'04), pages 640–645, Dalian, China, 2004.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. In Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM), pages 435–440, Montreal, Canada, 2012. ACL.

Marco Baroni and Silvia Bernardini. A new approach to the study of translationese:

Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274, 2006.

Alberto Barrón-Cedeño. On the Mono and Cross-Language Detection of Text Reuse and Plagiarism. *Universitat Politècnica de València*, 2012.

Alberto Barrón-Cedeño and Paolo Rosso. Towards the Exploitation of Statistical Language Models for Plagiarism Detection with Reference. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, *Proceedings of the18th European Conference on Artificial Intelligence (ECAI'08) Workshop on Uncovering Plagiarism, Authorship and Software Misuse (PAN'08)*, pages 15–19, Patras, Greece, 2008.

Alberto Barrón-Cedeño and Paolo Rosso. On Automatic Plagiarism Detection based on n-grams Comparison. In *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval (ECIR '09)*, pages 696–700, Toulouse, France, 2009. Springer.

Alberto Barrón-Cedeño and Paolo Rosso. Towards the 2nd International Competition on Plagiarism Detection and Beyond. In *Proceedings of the 4th International Plagiarism Conference*, volume 03, pages 1–12, Newcastle, UK, 2010.

Alberto Barrón-Cedeño, Paolo Rosso, David Pinto, and Alfons Juan. On cross-lingual plagiarism analysis using a statistical model. In *Proceedings of* the18th European Conference on Artificial Intelligence (ECAI'08) Workshop on Uncovering Plagiarism, Authorship and Software Misuse (PAN'08), 2008.

Alberto Barrón-Cedeño, Marta Vila, M. Antonia Martí, and Paolo Rosso. Plagiarism meets Paraphrasing: Insights for the Next Generation in Automatic Plagiarism Detection. Computational Linguistics, 39(4), 2013.

Charles Bird. The Detection of Cheating in Objective Examinations. School and Society, 25(-):261–262, 1927.

Stephan Bloehdorn and Alessandro Moschitti. Combined syntactic and semantic kernels for text classification. In Proceedings of the 29th European conference on Information Retrieval Research (ECIR'07), pages 307–318, Rome, Italy, 2007.

Sergey Brin, James Davis, and Hector Garcia-Molina. Copy Detection Mechanisms for Digital Documents. In Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '95), pages 398–409, San Jose, California, USA, 1995.

Joanna Bull, Carol Collins, Elisabeth Coughlin, and Dale Sharp. Technical review of plagiarism detection software report. Technical report, University of Luton, London, UK, 2001.

Zdenek Ceska. The Future of Copy Detection Techniques. In Proceedings of the 1st Young Researchers Conference on Applied Sciences, number 1995, pages 5–10, Pilsen, Czech Republic, 2007.

Zdenek Ceska. Automatic Plagiarism Detection Based on Latent Semantic Analysis. Doctoral thesis, University of West Bohemia, 2009.

Zdenek Ceska and Chris Fox. The Influence of Text Pre-processing on Plagiarism Detection. In Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP'09), pages 55–59, Borovets, Bulgaria, 2009.

Zdenek Ceska, Michal Toman, and Karel Jezek. Multilingual plagiarism detection. In Proceedings of the 13th international conference on Artificial Intelligence: Methodology, Systems, and Applications (AIMSA '08), pages 83–92, Varna, Bulgaria, 2008.

Jason S. Chang and Yu-Chia Chang. Computer Assisted Language Learning Based on Corpora and Natural Language Processing : The Experience of Project CANDLE. In Proceedings of the Interactive Workshop on Language e-Learning (IWLeL 2004), pages 15–23, 2004.

Chien-Ying Chen, Jen-Yuan Yeh, and Hao-Ren Ke. Plagiarism Detection using ROUGE and WordNet. Journal of Computing, 2(3):34–44, March 2010.

Gill Chester. Pilot of free-text electronic plagiarism detection software. Technical report, Joint Information Systems Committee (JISC), London, UK, 2001.

Miranda Chong and Lucia Specia. Lexical Generalisation for Word-level Matching in Plagiarism Detection. In Proceedings of the International Conference Recent

Advances in Natural Language Processing (RANLP'11), pages 704–709, Hissar, Bulgaria, 2011.

Miranda Chong and Lucia Specia. Linguistic and Statistical Traits Characterising Plagiarism. In Proceedings of the 24th International Conference on Computational Linguistics (COLING2012), volume 2, pages 195–204, Mumbai, India, 2012.

Miranda Chong, Lucia Specia, and Ruslan Mitkov. Using Natural Language Processing for Automatic Detection of Plagiarism. In Proceedings of the 4th International Plagiarism Conference, Newcastle, UK, 2010.

Daniela Chuda and Pavol Navrat. Support for checking plagiarism in e-learning. Procedia - Social and Behavioral Sciences, 2(2):3140–3144, 2010.

Paul Clough. Plagiarism in natural and programming languages: an overview of current tools and technologies. Technical report, University of Sheffield, Sheffield, UK, 2000.

Paul Clough. Old and new challenges in automatic plagiarism detection. National Plagiarism Advisory Service, (February):1–14, 2003.

Paul Clough and Mark Stevenson. Developing a corpus of plagiarised short answers. Language Resources and Evaluation, 45(1):5–24, 2010.

Paul Clough, Robert Gaizauskas, Scott Piao, and Yorick Wilks. METER: MEasuring TExt Reuse. In Proceedings of the 40th Annual Meeting on Association for

Computational Linguistics (ACL2002), pages 152–159, Philadelphia, PA, USA, 2002a. Association for Computational Linguistics.

Paul Clough, Robert Gaizauskas, and Scott S L Piao. Building and Annotating a Corpus for the Study of Journalistic Text Reuse. In: 3rd International Conference on Language Resources and Evaluation (LREC, pages 1678–1691, 2002b.

Courtney Corley and Rada Mihalcea. Measuring the semantic similarity of texts. In Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, number June, pages 13–18, Ann Arbor, Michigan, USA, 2005.

Rosa Maria Coyotl-Morales, Luis Villaseñor Pineda, Manuel Montes-y Gómez, and Paolo Rosso. Authorship attribution using word sequences. In Proceedings of the 11th Iberoamerican conference on Progress in Pattern Recognition, Image Analysis and Applications (CIARP'06), pages 844–853, Cancun , Mexico, 2006. Springer.

Eugene Creswick, Emi Fujioka, and Terrance Goan. Pedigree Tracking in the Face of Ancillary Content. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 18th European Conference on Artificial Intelligence (ECAI'08) Workshop on Uncovering Plagiarism, Authorship and Software Misuse (PAN'08), pages 21–25, Patras, Greece, 2008.

Fintan Culwin and Thomas Lancaster. Visualising intra-corpal plagiarism. In

Proceedings of the Fifth International Conference on Information Visualisation, pages 289–296, London, UK, 2001. IEEE Comput. Soc.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL Recognising Textual Entailment Challenge. *Machine Learning Challenges. Lecture Notes in Computer Science.*, 3944:177–190, 2006.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the fifth international conference on Language Resources and Evaluation(LREC2006)*, pages 449–454, Genoa, Italy, 2006. ELRA.

Rodolfo Delmonte, Sara Tonelli, Marco Boniforti, Antonella Bristot, and Emanuele Pianta. Venses - a linguistically-based system for semantic evaluation. In *Proceedings of the 1st International Conference on Machine Learning Challenges: evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment (MLCW'05)*, pages 49–52, Southampton, UK, 2005.

Michael Denkowski and Alon Lavie. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the 6th Workshop on Statistical Machine Translation (WMT '11)*, pages 85–91, Edinburgh, UK, 2011.

Heinz Dreher. Automatic Conceptual Analysis for Plagiarism Detection. *Issues in Informing Science and Information Technology*, 4:601–614, 2007.

Robert Evans. Evaluating an electronic plagiarism detection service: The importance of trust and the difficulty of proving students don't cheat. Active Learning in Higher Education, 7(1):87–99, March 2006.

Ol'ga Feiguina and Graeme Hirst. Authorship Attribution for Small Texts: Literary and Forensic Experiments. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 19–22, Amsterdam, Netherlands, 2007. ACM.

Karen Fullam and Jisun Park. Improvements for scalable and accurate plagiarism detection in digital documents. Data Mining and Knowledge Discovery, 7:1–14, 2002.

Robert Gaizauskas, Jonathan Foster, Yorick Wilks, John Arundel, Paul Clough, and Scott S L Piao. The METER Corpus: A corpus for analysing journalistic text reuse. In Proceedings of the Corpus Linguistics Conference, number 0114, pages 214–223, Lancaster, UK, 2001. University of Sheffield.

Martin Gellerstam. Translationese in Swedish novels translated from English. Lund: CWK Gleerup, 1986.

Jack William Grieve. Quantitative authorship attribution: A history and an evaluation of techniques. Master's thesis, Simon Fraser University, pages 1–150, 2002.

Cristian Grozea and Marius Popescu. Who's the Thief ? Automatic Detection of the Direction of Plagiarism. In Proceedings of the 11th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing'10), volume 6008, pages 700–710, Iai, Romania, 2010.

Cristian Grozea, Christian Gehl, and Marius Popescu. ENCOPLOT: Pairwise Sequence matching in Linear Time Applied to Plagiarism Detection. In Benno Stein, Paolo Rosso, Efstathios Stamatatos, Moshe Koppel, and Eneko Agirre, editors, Proceedings of the 25th Annual Conference of the Spanish Society for Natural Language Processing (SEPLN 2009), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'09), pages 10–18, Donostia, Spain, 2009.

Stefan Gruner and Stuart Naven. Tool support for plagiarism detection in text documents. In Proceedings of the 2005 ACM symposium on Applied computing (SAC '05), pages 776–781, Santa Fe, USA, 2005. ACM.

Hans Van Halteren. Source language markers in EUROPARL translations. In Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008), number 8, pages 937–944, Manchester, UK, 2008.

Christian Hardmeier. Improving Machine Translation Quality Prediction with Syntactic Tree Kernels. In Proceedings of the 15th Conference of the European Association for Machine Translation (EMAT-2011), number 5, pages 233–240, Leuven, Belgium, 2011.

Kenneth Heafield. KenLM: Faster and smaller language model queries.

In Proceedings of the 6th Workshop on Statistical Machine Translation (EMNLP2011), number 2009, pages 187–197, Edinburgh, UK, 2011.

Steffen Hedegaard and JG Simonsen. Lost in translation: authorship attribution using frame semantics. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL'11), pages 65–70, Portland, USA, 2011.

Timothy Hoad and Justin Zobel. Methods for identifying versioned and plagiarized documents. Journal of the American Society for Information Science and Technology, 54(3):203–215, 2003.

Iustina Ilisei and Diana Inkpen. Translationese Traits in Romanian Newspapers: A Machine Learning Approach. International Journal of Computational Linguistics and Applications, 2, 2011.

Iustina Ilisei, Diana Inkpen, Gloria Corpas Pastor, and Ruslan Mitkov. Identification of translationese: A machine learning approach. In Proceedings of the 11th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2010), pages 503–511, Iai, Romania, 2010.

Thorsten Joachims. Training linear SVMs in linear time. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (KDD '06), pages 217–226, Philadelphia, USA, 2006. ACM Press.

Patrick Juola, John Sofko, and Patrick Brennan. A Prototype for Authorship

Attribution Studies. Literary and Linguistic Computing, 21(2):169–178, April 2006.

Nam Oh Kang, Alexander Gelbukh, and Sang Yong Han. Ppchecker: Plagiarism pattern checker in document copy detection. In Proceedings of the 9th International Conference on Text, Speech and Dialogue (TSD'06), volume 4188, pages 661–668, Brno, Czech Republic, 2006. Springer.

Jussi Karlgren and Gunnar Eriksson. Authors, Genre, and Linguistic Convention. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 23–28, Amsterdam, Netherlands, 2007. ACM.

Jan Kasprzak and Michal Brandejs. Improving the Reliability of the Plagiarism Detection System. In Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2010), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'10), Padua, Italy, 2010.

Vlado Keselj, Fuchun Peng, Nick Cercone, and Calvin Thomas. N-gram-based author profiles for authorship attribution. In Proceedings of the Conference Pacific Association for Computational Linguistics (PACLING'03), volume 3, pages 255–264, Halifax, Canada, 2003. Citeseer.

Dan Klein and Christopher Manning. Accurate unlexicalized parsing. In

Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL'03), pages 423–430, Sapporo, Japan, 2003.

Katrin Kohler and Debora Weber-Wulff. Plagiarism Detection Test 2010. Technical report, HTW Berlin, 2010.

Moshe Koppel and Noam Ordan. Translationese and its dialects. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT'11), volume 1, pages 1318–1326, Portland, USA, 2011.

Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In Proceedings of the 21st International Conference on Machine learning (ICML'04), pages 62–68, New York, USA, 2004. ACM Press.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Computational methods in authorship attribution. Journal of the American Society for Information Science and Technology, 60(1):9–26, January 2009.

S. B. Kotsiantis. Supervised Machine Learning: A Review of Classification Techniques. In Proceedings of the 2007 conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies, pages 3–24. IOS Press Amsterdam, 2007.

David Kurokawa, Cyril Goutte, and Pierre Isabelle. Automatic Detection of Translated Text and its Impact on Machine Translation. In Proceedings of the 12th Machine Translation Summit (MT Summit XII), Ottawa, Canada, 2009.

Thomas Lancaster and Fintan Culwin. Classifications of plagiarism detection engines. Innovation in Teaching And Learning in Information and Computer Sciences (ITALICS), 4(2), 2003.

Thomas Lancaster and Fintan Culwin. A Visual Argument for Plagiarism Detection using Word Pairs. In Proceedings of the 1st International Plagiarism Conference, number 4, pages 1–14, Newcastle, UK, 2004a.

Thomas Lancaster and Fintan Culwin. Using freely available tools to produce a partially automated plagiarism detection process. In Proceedings of the 21st Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE'04), pages 520–529, Perth, Australia, 2004b.

Peter Lane, Caroline Lyon, and James Malcolm. Demonstration of the Ferret plagiarism detector. In Proceedings of the 2nd International Plagiarism Conference, Newcastle, UK, 2006.

Ann Lathrop and Kathleen Foss. Student Cheating and Plagiarism in the Internet Era. book, 2000.

Thomas Lavergne, Tanguy Urvoy, and Francois Yvon. Detecting Fake Content with Relative Entropy Scoring. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 18th European Conference on Artificial

Intelligence (ECAI'08) Workshop on Uncovering Plagiarism, Authorship and Software Misuse (PAN'08), pages 27–31, Patras, Greece, 2008.

Gennadi Lembersky, Noam Ordan, and Shuly Wintner. Language models for machine translation: original vs. translated texts. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011), pages 363–374, Edinburgh, Scotland, 2011.

Gennadi Lembersky, Noam Ordan, and Shuly Wintner. Adapting Translation Models to Translationese Improves SMT. In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012), pages 255–265, Avignon, France, 2012.

Chi-Hong Leung and Yuen-Yan Chan. A natural language processing approach to automatic plagiarism detection. In Procedding of the 8th ACM SIG-information conference on Information technology education (SIGITE '07), pages 213–218, Destin, USA, 2007. ACM Press.

Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004), pages 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics.

Romans Lukashenko, Vita Graudina, and Janis Grundspenkis. Computer-Based Plagiarism Detection Methods and Tools: An Overview. In Proceedings of the International Conference on Computer Systems and Technologies (CompSysTech'07), pages 1–6, Rousse, Bulgaria, 2007.

Caroline Lyon, James Malcolm, and Bob Dickerson. Detecting short passages of similar text in large document collections. In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2001), pages 118–125, Cornell, USA, 2001.

Caroline Lyon, Ruth Barrett, and James Malcolm. Plagiarism is easy, but also easy to detect. Plagiary: Cross-Disciplinary Studies in Plagiarism, Fabrication, and Falsification, 1(1):57–65, 2006.

Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma. Detecting near-duplicates for web crawling. In Proceedings of the 16th international conference on World Wide Web (WWW'07), pages 141–150, Banff, Canada, 2007.

Bill Marsh. Turnitin.com and the scriptural enterprise of plagiarism detection. Computers and Composition, 21(4):427–438, 2004.

Hermann Maurer and Bilal Zaka. Plagiarism - a problem and how to fight it. In Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications (ED-MEDIA), pages 25–28, Chesapeake, USA, 2007.

Hermann Maurer, Frank Kappe, and Bilal Zaka. Plagiarism - a survey. Journal of Universal Computer Science, 12(8):1050–1084, 2006.

Sven Meyer zu Eissen and Benno Stein. Intrinsic plagiarism detection. In

Proceedings of the 28th European Conference on IR Research (ECIR 2006), pages 565–569, London, UK, 2006. Springer.

George Mikros and Eleni Argiri. Investigating Topic Influence in Authorship Attribution. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 29–35, Amsterdam, Netherlands, 2007. ACM.

Tom M Mitchell. Machine Learning and Data Mining Over the past. Communications of the ACM, 42(11):30–36, 1999.

Ruslan Mitkov and Gloria Corpas Pastor. Improving Third Generation Translation Memory systems through identification of rhetorical predicates. In Proceedings of LangTech2008, Rome, Italy, 2008.

Krisztián Monostori, Arkdy Zaslavsky, and Heinz Schmidt. Document overlap detection system for distributed digital libraries. In Proceedings of the 5th ACM conference on Digital libraries (DL 00), pages 226–227, New York, USA, 2000. ACM Press.

Krisztián Monostori, Raphael Finkel, Arkady Zaslavsky, Gábor Hodász, and Máté Pataki. Comparison of overlap detection techniques. In Proceedings of the International Conference on Computational Science, number I, pages 51–60, Amsterdam, Netherlands, 2002. Springer.

Alessandro Moschitti. Making tree kernels practical for natural language learning. In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, pages 113–120, Trento, Italy, 2006a.

Alessandro Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In Proceedings of the 17th European conference on Machine Learning (ECML'06), pages 318–329, Berlin, Germany, 2006b.

Alessandro Moschitti, Bonaventura Coppola, and Daniele Pighin. Semantic Tree Kernels to classify Predicate Argument Structures. In Proceedings of the 17th European Conference on Artificial Intelligence (ECAI 2006), pages 568–572, Riva del Garda, Italy, 2006.

Maxim Mozgovoy. Enhancing computer-aided plagiarism detection. Academic Dissertation. University of Joensuu, 2007.

Maxim Mozgovoy, Vitaly Tusov, and Vitaly Klyuev. The Use of Machine Semantic Analysis in Plagiarism Detection. In Proceedings of the 9th International Conference on Humans and Computers, pages 72–77, Aizu-Wakamatsu, Japan, 2006.

Maxim Mozgovoy, Tuomo Kakkonen, and Erkki Sutinen. Using natural language parsers in plagiarism detection. In Proceedings of the Workshop on Spoken Language Technology for Education Workshop (SLaTE' 07), pages 7–9, Farmington, USA, 2007.

Markus Muhr, Roman Kern, Mario Zechner, and Michael Granitzer. External and

Intrinsic Plagiarism Detection using a Cross-Lingual Retrieval and Segmentation System Lab Report for PAN at CLEF 2010. In Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2010), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'10), Padua, Italy, 2010.

Dragos Stefan Munteanu and Daniel Marcu. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. Computational Linguistics, 31 (4):477–504, December 2005.

Thade Nahnsen, Ozlem Uzuner, and Boris Katz. Lexical chains and sliding locality windows in content-based text similarity detection. In Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP'05), pages 150–154, Jeju Island, Korea, 2005.

Rao Muhammad Adeel Nawab, Mark Stevenson, and Paul Clough. University of Sheffield Lab Report for PAN at CLEF 2010. In Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2010), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'10), Padua, Italy, 2010.

Rogelio Nazar and Marta Sánchez Pol. An Extremely Simple Authorship Attribution System. In Proceedings of the Second European IAFL Conference on Forensic Linguistics / Language and the Law, Barcelona, Spain, 2006.

Karl J. Ottenstein. An algorithmic approach to the detection and prevention of plagiarism. Newsletter ACM SIGCSE Bulletin, 8(4):30–41, 1976.

Kevin R Parker, Robert Williams, Philip S Nitse, and Albert S M Tay. Use of the Normalized Word Vector Approach in Document Classification for an LKMC. Journal of Issues in Informing Science and Information Technology, 5:513–524, 2008.

Gloria Corpas Pastor, Ruslan Mitkov, Naveed Afzal, and Lisette Garcia Moya. Translation universals: do they exist? A corpus-based and NLP approach to convergence. In Proceedings of the LREC2008 Workshop on Building and Using Comparable Corpora, Marrakech, Morocco, 2008.

Máté Pataki. Distributed Similarity and Plagiarism Search. In Proceedings of the Automation and Applied Computer Science Workshop (AACS2006), pages 121–130, Budapest, Hungary, 2006.

Alex Penev and Raymond Wong. Shallow NLP techniques for internet search. In Proceedings of the 29th Australasian Computer Science Conference (ACSC '06), volume 48, pages 167–176, Tasmania, Australia, 2006.

Maria Soledad Pera and Yiu-kai Ng. SimPaD : A Word-Similarity Sentence-Based Plagiarism Detection Tool on Web Documents. Web Intelligence and Agent Systems: An International Journal, 0:1–15, 2009.

Scott S L Piao, Paul Clough, and John Arundel. Proposing Basic Approaches to

Detecting Text Rewrite. Technical report, Department of Computer Science, University of Sheffield, Sheffield, UK, 2001.

David Pinto, Jorge Civera, Alberto Barrón-Cedeño, Alfons Juan, and Paolo Rosso. A statistical approach to crosslingual natural language tasks. Journal of Algorithms, 64(1):51–60, January 2009.

Martin Potthast, Alberto Barrón-Cedeño, Benno Stein, and Paolo Rosso. Cross-language plagiarism detection. Language Resources and Evaluation, 45(1):45–62, January 2010a.

Martin Potthast, Benno Stein, Alberto Barrón-Cedeño, and Paolo Rosso. An evaluation framework for plagiarism detection. In Proceedings of the 23rd International Conference on Computational Linguistics (COLING'10), pages 997–1005, Beijing, China, 2010b. Association for Computational Linguistics.

Martin Potthast, Benno Stein, Andreas Eiselt, Alberto Barrón-Cedeño, and Paolo Rosso. Overview of the 2nd International Competition on Plagiarism Detection. In Benno Stein, Paolo Rosso, Efstathios Stamatatos, Moshe Koppel, and Eneko Agirre, editors, Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2010), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'10), pages 1–9, Padua, Italy, 2010c.

Martin Potthast, Tim Gollub, Matthias Hagen, Jan Graß egger, Johannes Kiesel, Maximilian Michel, Arnd Oberländer, Martin Tippmann, Alberto Barrón-

Cedeño, Parth Gupta, Paolo Rosso, and Benno Stein. Overview of the 4th International Competition on Plagiarism Detection. In Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2012), Evaluation Labs and Workshop - Working Notes Papers (PAN'12), Rome, Italy, 2012.

Bruno Pouliquen, Ralf Steinberger, and Camelia Ignat. Automatic identification of document translations in large multilingual document collections. In Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP'03), number 2002, pages 401–408, Borovets, Bulgaria, 2003.

Radim Rehurek. Semantic-based plagiarism detection. Technical report, Faculty of Informatics, Masaryk University, Brno, Czech Republic, 2007.

Matthias Robine, Pierre Hanna, Pascal Ferraro, and Julien Allali. Adaptation of String Matching Algorithms for Identificaton of Near-Duplicate Music Documents. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 37–43, Amsterdam, Netherlands, 2007. ACM.

Per Runeson, Magnus Alexandersson, and Oskar Nyholm. Detection of Duplicate Defect Reports Using Natural Language Processing. In Proceedings of the 29th

International Conference on Software Engineering (ICSE'07), pages 499–510, Minneapolis, USA, May 2007. Ieee.

Chang-keon Ryu, Hyong-jun Kim, Seung-hyun Ji, Gyun Woo, and Hwan-gue Cho. Detecting and tracing plagiarized documents by reconstruction plagiarism-evolution tree. In Proceedings of the 8th IEEE International Conference on Computer and Information Technology (CIT 2008), pages 119–124, Sydney, Australia, July 2008. Ieee.

Fernando Sánchez-Vega, Esaú Villatoro-Tello, Manuel Montes-y Gómez, Luis Villaseñor Pineda, and Paolo Rosso. Determining and characterizing the reused text for plagiarism detection. Expert Systems with Applications, 40(5):1804–1813, April 2013.

Narayanan Shivakumar and Hector Garcia-Molina. SCAM: A copy detection mechanism for digital documents. In Proceedings of the Second Annual Conference on the Theory and Practice of Digital Libraries, pages 1–13, Texas, USA, 1995.

Narayanan Shivakumar and Hector Garcia-Molina. Building a scalable and accurate copy detection mechanism. In Proceedings of 1st ACM International Conference on Digital Libraries (DL'96), pages 160–168, Bethesda, USA, 1996.

Daria Sorokina, Johannes Gehrke, Simeon Warner, and Paul Ginsparg. Plagiarism detection in arXiv. In Proceedings of the Sixth International Conference on Data Mining (ICDM 06), number 7, pages 1070–1075, Hong Kong, 2006. IEEE Computer Society Washington, DC, USA.

Rui Sousa-Silva, Tim Grant, and Belinda Maia. "I didn't mean to steal someone else's words!": A Forensic Linguistic Approach to Detecting Intentional Plagiarism. In Proceeding of the 4th International Plagiarism Conference, pages 1–31, Newcastle, UK, 2010.

Efstathios Stamatatos. The Class Imbalance Problem in Author Identification. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), page 9, Amsterdam, 2007. ACM.

Efstathios Stamatatos. A Survey of Modern Authorship Attribution Methods. Journal of the American Society for Information Science, 60(3):538–556, 2009.

Benno Stein and Sven Meyer zu Eissen. Intrinsic Plagiarism Analysis with Meta Learning. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 45–50, Amsterdam, Netherlands, 2007. ACM.

Benno Stein, Moshe Koppel, and Efstathios Stamatatos. Plagiarism Analysis, Authorship Identification, and Near-Duplicate Detection. In Benno Stein, Moshe Koppel, and Efstathios Stamatatos, editors, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), Amsterdam, Netherlands, 2007a.

Benno Stein, Sven Meyer zu Eissen, and Martin Potthast. Strategies for retrieving plagiarized documents. In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), pages 825–826, New York, USA, 2007b. ACM Press. ISBN 9781595935977.

Benno Stein, Nedim Lipka, and Sven Meyer zu Eissen. Meta Analysis within Authorship Verification. In Proceedings of the19th International Conference on Database and Expert Systems Application (DEXA 2008), pages 34–39, Turin, Italy, 2008a. IEEE Computer Society Washington, DC, USA.

Benno Stein, Efstathios Stamatatos, and Moshe Koppel. Uncovering Plagiarism, Authorship and Social Software Misuse - PAN'08. In Benno Stein, Efstathios Stamatatos, and Moshe Koppel, editors, Proceedings of the18th European Conference on Artificial Intelligence (ECAI'08) Workshop on Uncovering Plagiarism, Authorship and Software Misuse (PAN'08), Patras, Greece, 2008b.

Benno Stein, Paolo Rosso, Efstathios Stamatatos, Moshe Koppel, and Eneko Agirre. 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse. In Benno Stein, Paolo Rosso, Efstathios Stamatatos, Moshe Koppel, and Eneko Agirre, editors, Proceedings of the 25th Annual Conference of the Spanish Society for Natural Language Processing (SEPLN 2009), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'09), Donostia, Spain, 2009.

Andreas Stolcke. SRILM-an extensible language modeling toolkit. In Proceedings of the 7th International Conference on Spoken Language Processing, volume 3, pages 901–904, Denver, USA, 2002.

Takashi Tashiro, Takanori Ueda, Taisuke Hori, Yu Hirate, and Hayato Yamana. EPCI: extracting potentially copyright infringement texts from the web. In Proceedings of the 16th international conference on World Wide Web (WWW'07), pages 1151–1152, Banff, Canada, 2007. ACM.

Rafael M. Terol, Patricio Martínez-Barco, and Manuel Palomar. An Architecture for Spoken Document Retrieval. In Proceeding of the 7th International conference on Text, Speech and Dialogue (TSD2004), volume 3206, pages 505–511, Brno, Czech Republic, 2004.

Roman Tesar, Massimo Poesio, Vaclav Strnad, and Karel Jezek. Extending the single words-based document model: a comparison of bigrams and 2-itemsets. In Proceedings of the 2006 ACM symposium on Document engineering, pages 138–146, Amsterdam, Netherlands, 2006. ACM.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pages 173–180, Edmonton, Canada, 2003.

George Tsatsaronis, Iraklis Varlamis, Andreas Giannakoulopoulos, and Nikolaos Kanellopoulos. Identifying free text plagiarism based on semantic similarity.

In Proceedings of the 4th International Plagiarism Conference, Newcastle, UK, 2010.

Özlem Uzuner, Boris Katz, and Thade Nahnsen. Using syntactic information to identify plagiarism. In Proceedings of the second workshop on Building Educational Applications Using NLP (EdAppsNLP'05), pages 37–44, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

Vered Volansky, Noam Ordan, and Shuly Wintner. More Human or more Translated? Original Texts vs. Human and Machine Translations. In Proceedings of the Israeli Seminar on Computational Linguistics (ISCOL2011), Bar-Ilan, Israel, 2011.

Debora Weber-Wulff. On the utility of plagiarism detection software. In 3rd International Plagiarism Conference, Newcastle upon Tyne, number September 2007, pages 1–11, Newcastle, UK, 2008.

Debora Weber-Wulff. Test cases for plagiarism detection software. In Proceedings of the 4th International Plagiarism Conference, pages 1–13, Newcastle, UK, 2010.

Daniel R. White and Mike S. Joy. Sentence-based natural language plagiarism detection. Journal on Educational Resources in Computing, 4(4):1–20, December 2004.

Jeremy Williams. The plagiarism problem: are students entirely to blame? In Proceedings of the 19th Annual Conference of the Australasian Society for

Computers in Learning in Tertiary Education (ASCILITE), volume 2, pages 721–730, Auckland, New Zealand, 2002.

Robert Williams. The power of normalised word vectors for automatically grading essays. Journal of Issues in Informing Science and Information Technology Volume, 3:721–730, 2006.

Michael Wise. Running karp-rabin matching and greedy string tiling. Technical Report 463, Basser Department of Computer Science, University of Sydney, Sydney, Australia, 1993.

Michael Wise. YAP3: improved detection of similarities in computer program and other texts. In SIGCSE '96: Proceedings of the 27th SIGCSE technical symposium on Computer science education, volume 28, pages 130–134, Philadelphia, USA, 1996.

Chuan Xiao, Wei Wang, Xuemin Lin, and Jeffery Xu Yu. Efficient similarity joins for near duplicate detection. In Proceedings of the 17th international conference on World Wide Web (WWW'08), pages 131–140, Bejing, China, 2008. ACM New York, NY, USA.

Hui Yang and Jamie Callan. Near-duplicate detection by instance-level constrained clustering. Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '06), pages 421–428, 2006.

Shiqi Zhao, Haifeng Wang, Ting Liu, and Sheng Li. Extracting paraphrase patterns from bilingual parallel corpora. Natural Language Engineering, 15(04):503, 2009.

R. Zheng, Jiexun Li, Hsinchun Chen, and Zan Huang. A framework for authorship identification of online messages: Writing-style features and classification techniques. Journal of the American Society for Information Science and Technology, 57(3):378–393, 2006.

Manuel Zini, Marco Fabbri, Massimo Moneglia, and Alessandro Panunzi. Plagiarism Detection through Multilevel Text Comparison. 2006 Second International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution (AXMEDIS'06), pages 181–185, December 2006. doi: 10.1109/AXMEDIS.2006.40.

Du Zou, Wei-jiang Long, and Zhang Ling. A Cluster-Based Plagiarism Detection Method. In Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF 2010), Uncovering Plagiarism, Authorship, and Social Software Misuse Worksop (PAN'10), Padua, Italy, 2010.

Appendix A

List of Publications

1. Chong, M., Specia, L. & Mitkov, R. (2010) "Using Natural Language Processing for Automatic Detection of Plagiarism". In: Proceedings of the 4th International Plagiarism Conference (IPC2010), Newcastle-upon-Tyne, UK.
2. Chong, M., Specia, L. (2011) "Lexical Generalisation for Word-level Matching in Plagiarism Detection". In: Proceedings of the International Conference Recent Advances in Natural Language Processing 2011 (RANLP2011), pp. 704-709, Hissar, Bulgaria.
3. Chong, M., Specia, L. (2012) "Linguistic and Statistical Traits Characterising Plagiarism". In: Proceedings of the 24th International Conference on Computational Linguistics (COLING2012), Mumbai, India.

Appendix B

Additional Information on the Small-scale Experiment

B.1 Results

Table B.1 shows the correlations and accuracies of individual features and metrics.

Figure B.1 visualises the performance of individual features.

Feature	Technique	Metric	Correlation	Accuracy	Note
1	Original baseline	3-gram Jaccard (Ferret)	0.631	66.32%	
2	Sentence Segmentation	3-gram Jaccard (Ferret)	0.631	66.32%	
3	Tokenisation	3-gram Jaccard (Ferret)	0.631	66.32%	
4	Lowercasing	3-gram Jaccard (Ferret)	0.631	66.32%	
5	Tokenisation + Lowercasing	3-gram Jaccard (Ferret)	0.631	66.32%	
6	5 + Part-of-Speech Tagging	3-gram Jaccard (Ferret)	0.610	53.68%	
7	5+ Stopword Removal	3-gram Jaccard (Ferret)	0.618	63.16%	
8	5 + Lemmatisation	3-gram Jaccard (Ferret)	0.631	62.11%	* Baseline
9	7 + 8	3-gram Jaccard (Ferret)	0.621	63.16%	
10	7 + Stemming	3-gram Jaccard (Ferret)	0.625	65.26%	
11	7 + Punctuation Removal	3-gram Jaccard (Ferret)	0.617	64.21%	
12	11 + Part-of-Speech Tagging	3-gram Jaccard (Ferret)	0.591	56.84%	
13	11 + Number Replacement	3-gram Jaccard (Ferret)	0.617	64.21%	

14	12 + 13	3-gram Jaccard (Ferret)	0.602	57.89%	
15	Chunking	3-gram Jaccard (Ferret)	0.566	51.58%	
16	Out of Vocabulary Rate	Language Model	0.600	49.47%	Based on feature 5
17	1-gram Log Probability	Language Model	0.556	46.32%	
18	1-gram Log Probability without end of sentence mark	Language Model	0.472	45.26%	
19	1-gram Perplexity	Language Model	0.313	43.16%	
20	2-gram Log Probability	Language Model	0.384	44.21%	
21	2-gram Log Probability without end of sentence mark	Language Model	0.685	54.74%	
22	2-gram Perplexity	Language Model	0.669	54.74%	
23	3-gram Log Probability	Language Model	0.376	44.21%	

24	3-gram Log Probability without end of sentence mark	Language Model	0.688	53.68%		
25	3-gram Perplexity	Language Model	0.671	54.74%		
26	4-gram Log Probability (Chunking)	Language Model	0.141	40%		
27	4-gram Log Probability without end of sentence mark (Chunking)	Language Model	0.287	43.16%		
28	4-gram Perplexity (Chunking)	Language Model	0.280	42.11%		
29	5-gram Log Probability (Chunking)	Language Model	0.152	41.05%		
30	5-gram Log Probability without end of sentence mark (Chunking)	Language Model	0.281	42.11%		

31	5-gram Perplexity (Chunking)	Language Model	0.272	40%	
32	Out of Vocabulary Words: 4 & 5 grams	Longest Common Subsequence	0.476	45.26%	Based on feature 5
33	Match rate with Original Documents	Longest Common Subsequence	0.427	42.11%	
34	Match rate with Suspicious Documents	Longest Common Subsequence	0.283	38.95%	
35	Average Number of Word Matches	Longest Common Subsequence	0.312	41.05%	
36	Maximum Word Matches	Longest Common Subsequence	0.547	46.32%	
37	Ratio of Word Count	Longest Common Subsequence	0.186	36.84%	
38	Ratio of Sentence Count	Longest Common Subsequence	0.214	36.84%	

39	Total Number of Word Matches	Longest Common Subsequence	0.283	38.95%	
40	Dependency Relation Extraction	Containment Measure	0.760	69.47%	
41	Dependency Relation Extraction	Overlap Coefficient	0.654	60%	
42	Normalised Count by Number of Suspicious Sentences	Containment Measure	0.751	65.26%	Based on feature 5
43	Lexical Generalisation	1-gram Jaccard Coefficient	0.783	60%	*Synonyms matching
44	Original Baseline: Unique Overlap	Containment Measure	0.772	70.53%	Based on feature 5
45	Original Baseline: Non-unique Overlap	Containment Measure	0.655	61.05%	Based on feature 5
46	Original Baseline: 3-gram	Containment Measure	0.768	68.42%	Based on feature 5

47	Lemmatisation: Unique Overlap	Containment Measure	0.772	70.53%	Based on feature 8
48	Lemmatisation: Non-unique Overlap	Containment Measure	0.655	60%	Based on feature 8
49	Lemmatisation: 3-gram	Containment Measure	0.769	68.42%	Based on feature 8
50	Main Topic Raw Matches	1-gram Containment Measure	0.416	44.21%	*Extract topics from sentences
51	Main Topic Lemmatisation Matches	1-gram Containment Measure	0.388	49.47%	
52	Syntactic Constituent Raw Matches	1-gram Containment Measure	0.760	64.21%	
53	Syntactic Constituent: Punctuation Removal, Stopword Removal	1-gram Containment Measure	0.765	63.16%	

54	Syntactic Constituent: Lemmatisation	1-gram Containment Measure	0.758	63.16%	
55	FSyntactic Constituent: Lemmatisation, Stopword Removal	1-gram Containment Measure	0.768	65.26%	
56	Syntactic Constituent Remove Singleton	1-gram Containment Measure	0.731	61.05%	*Remove singleton constituents
Feature	**Technique**	**Metric**	**Correlation**	**Accuracy**	**Note**

Table B.1: Correlation Coefficient and Machine Learning Classifier Accuracy of Individual Features in the Small-scale Experiment

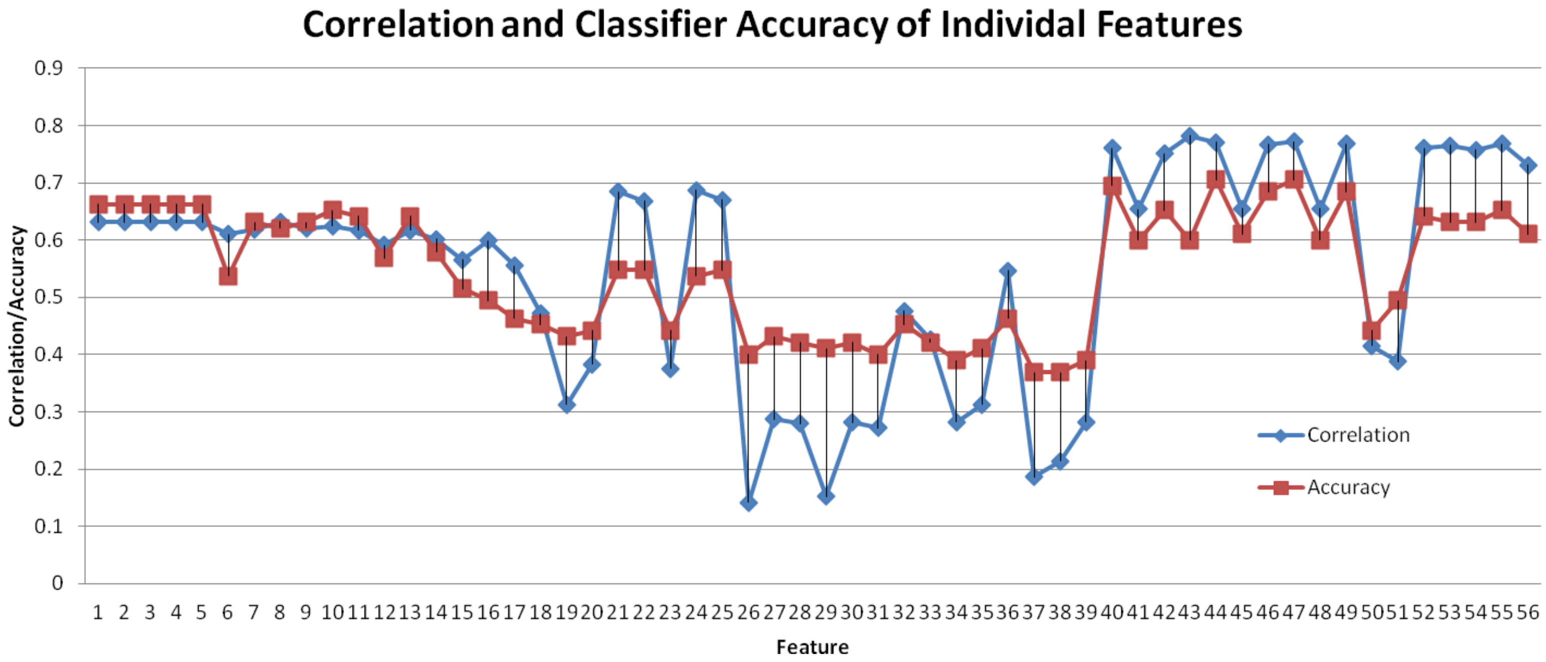

Figure B.1: Correlation and accuracy of the small-scale experiment

Appendix C

Additional Information on the Large-scale Experiment

Tables C.1 show the results of the individual features in the document-level initial experiment (Section 6.1). Table C.2 shows the results of the individual and combined features in the document-level additional experiment (Section 6.2).

Table C.3 lists the detailed results of the passage-level experiment in two classes (Section 6.3). Tables C.4 and C.5 show the overall results, and the performance of the individual and combined features in the passage-level experiment respectively (Section 6.3).

Feature	**True Positive**	**False Positive**	**False Negative**	**Precision**	**Recall**	**F-score**
1-gram overlap (Feature 1)	10	1,573	2,496	0.006	0.004	0.005
5-gram overlap (Feature 2)	1,280	1,256	1,226	0.505	0.511	0.508
5-gram sentence level (Feature 3)	1,364	238	1,142	0.852	0.544	0.664
5-gram document level (Feature 4)	1,363	171	1,143	0.889	0.544	0.675

Table C.1: Results of individual features in Section 6.1

Feature	**Accuracy**	**Micro-average**			**Class: Clean**			**Class: Plagiarised**			**Macro-average**		
		P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
1	91.52%	0.914	0.915	0.915	0.942	0.95	0.946	0.818	0.794	0.805	0.88	0.872	0.876
2	80.75%	0.786	0.808	0.784	0.832	0.943	0.884	0.623	0.33	0.432	0.728	0.637	0.658
3	77.87%	0.606	0.779	0.682	0.779	1	0.876	0	0	0	0.390	0.5	0.438
4	80.20%	0.78	0.802	0.782	0.834	0.931	0.88	0.589	0.349	0.438	0.712	0.64	0.659
5	83.70%	0.825	0.837	0.82	0.853	0.955	0.901	0.726	0.422	0.534	0.790	0.689	0.718
6	79.40%	0.764	0.794	0.751	0.809	0.962	0.879	0.604	0.201	0.302	0.707	0.582	0.591
1 + 2	91.52%	0.915	0.915	0.915	0.945	0.946	0.946	0.809	0.807	0.808	0.877	0.877	0.877
1 + 2 + 4 + 5 + 6	91.58%	0.916	0.916	0.916	0.945	0.947	0.946	0.811	0.808	0.809	0.878	0.878	0.878
1 + 2 + 4 + 5	91.61%	0.916	0.916	0.916	0.946	0.946	0.946	0.809	0.812	0.811	0.878	0.879	0.879
1 + 3	91.52%	0.914	0.915	0.915	0.942	0.95	0.946	0.818	0.794	0.805	0.88	0.872	0.876

1 + 4	91.54%	0.915	0.915	0.915	0.942	0.95	0.946	0.818	0.795	0.806	0.88	0.873	0.876
1 + 5	91.34%	0.915	0.913	0.914	0.95	0.938	0.944	0.791	0.827	0.809	0.871	0.883	0.877
1 + 6	91.50%	0.916	0.915	0.915	0.948	0.943	0.945	0.802	0.817	0.81	0.875	0.88	0.878
1 + 2 + 4 + 6	91.63%	0.915	0.916	0.916	0.942	0.951	0.947	0.822	0.794	0.808	0.882	0.873	0.878
All	91.65%	0.916	0.917	0.916	0.946	0.947	0.946	0.812	0.811	0.811	0.879	0.879	0.879
		P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
Feature	**Accuracy**	**Micro-average**			**Class: Clean**			**Class: Plagiarised**			**Macro-average**		

Table C.2: Results of the experiment on simulated plagiarism cases at the document level in Section 6.2

	Class:Clean					Class:Plagiarised				
Precision										
Split	**3-gram1**[41]	**Best Features**[42]	**PAN1**	**PAN2**	**3-gram2**[43]	**3-gram1**	**Best Features**	**PAN1**	**PAN2**	**3-gram2**
1	0.982	0.994	0.979	0.975	0.982	0.872	0.913	0.906	0.959	0.872
2	0.983	0.995	0.981	0.975	0.983	0.86	0.886	0.889	0.95	0.860
3	0.984	0.995	0.977	0.975	0.985	0.876	0.903	0.921	0.951	0.873
Avg.	0.982	0.995	0.979	0.975	0.983	0.877	0.9	0.905	0.953	0.868
Recall										
1	0.987	0.99	0.991	0.996	0.987	0.831	0.947	0.805	0.761	0.830
2	0.986	0.987	0.989	0.996	0.986	0.836	0.954	0.812	0.752	0.836
3	0.987	0.989	0.993	0.996	0.987	0.848	0.955	0.779	0.758	0.855
Avg.	0.988	0.989	0.991	0.996	0.986	0.83	0.951	0.802	0.757	0.840

[41] 3-gram1: With machine learning C4.5 10-fold cross-validation

[42] Best Features: With machine learning C4.5 10-fold cross-validation

[43] 3-gram2: Without machine learning

F-score

1	0.984	0.992	0.985	0.985	0.984	0.851	0.93	0.853	0.848	0.850
2	0.984	0.991	0.985	0.985	0.984	0.848	0.919	0.848	0.84	0.848
3	0.986	0.992	0.985	0.985	0.986	0.862	0.928	0.844	0.844	0.864
Avg.	0.985	0.992	0.985	0.985	0.985	0.853	0.925	0.85	0.844	0.854
Split	**3-gram1**	**Best Features**	**PAN1**	**PAN2**	**3-gram2**	**3-gram1**	**Best Features**	**PAN1**	**PAN2**	**3-gram2**

Table C.3: Comparative results of the experiment on simulated plagiarism cases at the passage level in Section 6.3, split into two classes

Overall

Precision

Split	**3-gram1**[44]	**Best Features**[45]	**PAN1**	**PAN2**	**3-gram2**[46]
1	0.927	0.954	0.943	0.967	0.927
2	0.922	0.941	0.935	0.963	0.922
3	0.93	0.949	0.949	0.963	0.929
Avg.	0.930	0.948	0.942	0.964	0.926

Recall

Split	**3-gram1**[44]	**Best Features**[45]	**PAN1**	**PAN2**	**3-gram2**[46]
1	0.909	0.969	0.898	0.879	0.909
2	0.911	0.971	0.901	0.874	0.911
3	0.918	0.972	0.886	0.877	0.921
Avg.	0.909	0.97	0.897	0.877	0.913

F-score

[44] 3-gram1: With machine learning C4.5 10-fold cross-validation

[45] Best Features: With machine learning C4.5 10-fold cross-validation

[46] 3-gram2: Without machine learning

Split	3-gram1	Best Features	PAN1	PAN2	3-gram2
1	0.918	0.961	0.919	0.917	0.917
2	0.916	0.955	0.917	0.913	0.916
3	0.924	0.96	0.915	0.915	0.925
Avg.	0.919	0.959	0.918	0.915	0.919

Accuracy

Split	3-gram1	Best Features	PAN1	PAN2	3-gram2
1	0.972	0.986	0.973	0.973	0.972
2	0.972	0.984	0.973	0.973	0.972
3	0.974	0.986	0.972	0.973	0.974
Avg.	0.973	0.985	0.973	0.973	0.973

Table C.4: Comparative results of the passpage level experiment on simulated casesin Section 6.3, overall

	Clean			**Plagiarised**			**Weighted Avg.**[47]			
Feature	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**Accuracy**
1	0.982	0.988	0.985	0.877	0.83	0.853	0.972	0.973	0.972	97.26%
2	0.964	0.977	0.97	0.749	0.658	0.7	0.944	0.946	0.945	94.61%
3	0.988	0.987	0.987	0.879	0.884	0.881	0.977	0.977	0.977	97.72%
4	0.93	0.991	0.959	0.769	0.296	0.427	0.915	0.924	0.908	92.41%
5	0.955	0.987	0.971	0.823	0.56	0.666	0.942	0.946	0.942	94.63%
6	0.973	0.989	0.981	0.875	0.739	0.801	0.963	0.965	0.964	96.49%
7	0.982	0.991	0.987	0.91	0.826	0.866	0.975	0.976	0.975	97.55%
All	0.995	0.989	0.992	0.899	0.949	0.923	0.985	0.985	0.985	98.49%
Best[48]	0.995	0.989	0.992	0.9	0.951	0.925	0.986	0.985	0.985	98.52%

[47] The weighted average is calculated where the clean class is more prominent than the plagiarism class.

[48] Best features: 1 + 2 + 3 + 6 + 7

Table C.5: Machine Learning results of individual and combined features of the experiment on simulated plagiarism cases at the passage level in Section 6.3

Appendix D

Additional Information on the Plagiarism Direction Experiment

Table D.1 describes the features used in the plagiarism direction experiment (Chapter 7). Table D.2 lists the InfoGain scores for the top ranking features across the three datasets (Section 7.3), and Table D.3 shows the detailed results of the experiment.

Feature	Technique	Note
1	Average Token Length	Total Characters / Total Tokens
2	Average Sentence Length	Total Tokens/ Total Sentences
3	Information Load	Lexical Words/ Total Tokens
4	Lexical Variety (Type/Token Rate)	Unique Word Types/Total Tokens
5	Lexical Richness	Unique Lemmatised Word Types/Total Tokens
6	Sentence without Finite Verbs	Sentence without finite verbs/ Total Sentences
7	Simple Sentence	Sentence contains one finite verbs/ Total Sentences
8	Complex Sentence	Sentence contains more than one finite verbs/ Total Sentences
9	Noun/Token Rate	Proportion of nouns over tokens
10	Preposition/Token Rate	Proportion of prepositions over tokens
11	Pronoun/Token Rate	Proportion of pronouns over tokens
12	Stopword/Token Rate	Proportion of stopwords over tokens
13	Finite Verb/Token Rate	Proportion of finite verbs over tokens

14	Grammatical Cohesion Rate	Grammatical Words/ Lexical Words
15	Individual Function Words	303 individual function words
16	Total Function Words/Token Rate	Total Function Words in List/ Tokens
17	Sentence Count	Number of sentences
18	Word Count	Number of word tokens
19	Character Count	Number of characters
20	3-gram Log Probability	Language Model Feature
21	3-gram Perplexity 1	Language model perplexity (all tokens)
22	3-gram Perplexity 2	Language model perplexity (without end of sentence tags)
23	5-gram Log Probability	Language Model Feature
24	5-gram Perplexity 1	Language model perplexity (all tokens)
25	5-gram Perplexity 2	Language model perplexity (without end of sentence tags)
26	Syntactic Tree	Parse trees generated from parsing

Feature	Technique	Note

Table D.1: Features used in the Plagiarism Direction Experiment

	Simulated		**Artificial**		**Non-parallel**	
InfoGain Rank	**Feature**	**Score**	**Feature**	**Score**	**Feature**	**Score**
1	19	0.197	20	0.821	15(be)	0.826
2	25	0.180	23	0.796	20	0.812
3	24	0.176	21	0.245	23	0.799
4	22	0.161	24	0.235	16	0.468
5	21	0.156	22	0.226	22	0.253
6	18	0.073	25	0.217	24	0.233
7	15(upon)	0.018	6	0.104	25	0.222
8	2	0.016	8	0.053	21	0.210
9	3	0.016	15(self)	0.030	15(and)	0.174
10	14	0.015	3	0.028	15(of)	0.135
11	13	0.015	15(here)	0.024	15(to)	0.114
12	15(of)	0.013	2	0.020	15(the)	0.102

13	15(which)	0.012	9	0.018	15(which)	0.084	
14	15(onto)	0.011	16	0.016	6	0.074	
15	15(because)	0.011	12	0.016	15(because)	0.058	
16	15(via)	0.010	15(other)	0.012	3	0.036	
17	15(be)	0.009	15(not)	0.012	8	0.030	
18	15(someone)	0.008	15(little)	0.011	15(someone)	0.020	
19	15(and)	0.008	17	0.011	2	0.018	
20	23	0.006	15(anywhere)	0.011	9	0.015	
21	20	0.006	5	0.009	12	0.012	
22	10	0.006	14	0.007	6	0.011	
23	6	0.005	4	0.007	17	0.010	
24	7	0.004	15(thus)	0.006	7	0.009	
25	12	0.004	15(me)	0.005	15(upon)	0.009	
26	16	0.004	7	0.004	14	0.004	
27	9	0.003	19	0.003	11	0.002	
28	15(the)	0.003	11	0.002	5	0.002	

29	1	0.002	—	—	19	0.002
30	17	0.001	—	—	—	—
InfoGain Rank	**Feature**	**Score**	**Feature**	**Score**	**Feature**	**Score**
	Simulated		**Artificial**		**Non-parallel**	

Table D.2: InfoGain attribute evaluator rankings for the three datasets in Section 7.3

Class	Feature set	P	R	F

Simulated

Class	Feature set	P	R	F
Original	All features	73.80%	76.40%	75.10%
	Pre-selected	75.80%	75.40%	75.60%
	Statistical	73.60%	75.50%	74.50%
	Simplification-based	59.90%	59.40%	59.70%
	Morphological	59.80%	58.20%	59.00%
Plagiarised	All features	75.60%	72.90%	74.20%
	Pre-selected	75.50%	75.90%	75.70%
	Statistical	74.80%	72.90%	73.80%
	Simplification-based	59.70%	60.20%	60%
	Morphological	59.30%	60.80%	60%
Average	All features	74.70%	74.70%	74.70%
	Pre-selected	75.65%	75.65%	75.65%
	Statistical	74.20%	74.20%	74.20%
	Simplification-based	59.80%	59.80%	59.80%
	Morphological	59.50%	59.50%	59.50%

Parallel Artificial

Class	Feature set	P	R	F
Original	All features	98.40%	97.90%	98.10%
	Pre-selected	98.40%	97.50%	97.90%
	Statistical	97.80%	97.70%	97.80%
	Simplification-based	67.80%	72.20%	72.20%

	Morphological	66.10%	74.10%	69.90%
Plagiarised	All features	97.90%	98.40%	98.10%
	Pre-selected	97.50%	98.40%	97.90%
	Statistical	97.70%	97.80%	97.80%
	Simplification-based	73.50%	63.30%	68%
	Morphological	70.50%	62.10%	66%
Average	All features	98.10%	98.10%	98.10%
	Pre-selected	97.95%	97.95%	97.90%
	Statistical	97.75%	97.75%	97.80%
	Simplification-based	70.65%	67.75%	70.10%
	Morphological	68.30%	68.10%	67.95%

Non-parallel Artificial

Original	All features	99.50%	99.50%	99.50%
	Pre-selected	98.5%	98.3%	98.4%
	Statistical	98.6%	98.2%	98.4%
	Simplification-based	69.7%	76.4%	72.9%
	Morphological	97.1%	97.7%	97.4%
Plagiarised	All features	99.50%	99.50%	99.50%
	Pre-selected	98.3%	98.5%	98.4%
	Statistical	98.2%	98.6%	98.4%
	Simplification-based	73.9%	66.8%	70.2%
	Morphological	97.7%	97.1%	97.4%

Average	All features	99.50%	99.50%	99.50%
	Pre-selected	98.40%	98.40%	98.40%
	Statistical	98.40%	98.40%	98.40%
	Simplification-based	71.80%	71.60%	71.55%
	Morphological	97.40%	97.40%	97.40%
Class	**Feature set**	**P**	**R**	**F**

Table D.3: Precision, recall and f-score of various features in the rule-based classifier

Appendix E

Additional Resources

Table E.1 shows the list of function words (stopwords) used in the experiments. Table E.2 lists the software tools and resources which were used in the study, and finally Section E.3 lists some real-life plagiarism incidents.

E.1 Function Words

a	abroad	about	above	across	after
again	against	ago	ahead	all	almost
alongside	already	also	although	always	am
amid	amidst	among	amongst	an	and
another	any	anybody	anyone	anything	anywhere
apart	are	aren’t	around	as	aside
at	away	back	backward	backwards	be
because	been	before	beforehand	behind	being
below	between	beyond	both	but	by
can	can’t	cannot	could	couldn’t	dare
daren’t	despite	did	didn’t	directly	do
does	doesn’t	doing	don’t	done	down

during	each	either	else	elsewhere	enough
even	ever	evermore	every	everybody	everyone
everything	everywhere	except	fairly	farther	few
fewer	for	forever	forward	from	further
furthermore	had	hadn’t	half	hardly	has
hasn’t	have	haven’t	having	he	hence
her	here	hers	herself	him	himself
his	how	however	if	in	indeed
inner	inside	instead	into	is	isn’t
it	its	itself	just	keep	kept
later	least	less	lest	like	likewise
little	low	lower	many	may	mayn’t
me	might	mightn’t	mine	minus	moreover
most	much	must	mustn’t	my	myself
near	need	needn’t	neither	neverf	neverless
next	no	no-one	nobody	none	nor
not	nothing	notwithstanding	now	nowhere	of
off	often	on	once	one	ones
only	onto	opposite	or	other	others
otherwise	ought	oughtn’t	our	ours	ourselves
out	outside	over	own	past	per
perhaps	please	plus	provided	quite	rather

really	round	same	self	selves	several
shall	shan't	she	should	shouldn't	since
so	some	somebody	someday	someone	something
sometimes	somewhat	still	such	than	the
their	theirs	them	themselves	then	there
therefore	these	they	thing	things	this
those	though	through	throughout	thus	till
to	together	too	towards	under	underneath
undoing	unless	unlike	until	up	upon
upwards	us	versus	very	via	was
wasn't	way	we	well	were	weren't
what	whatever	when	whence	whenever	where
whereby	wherein	wherever	whether	which	whichever
while	whilst	whither	who	whoever	whom
whose	why	will	with	within	without
won't	would	wouldn't	yet	you	your
yours	yourself	yourselves	—	—	—

Table E.1: List of function words used throughout the experiments

E.2 Software Tools

Tool	Description	Usage
	For feature extraction	
NLTK	Python NLP package	POS tagging, chunking, lemmatisation, stemming
http://nltk.org/		
Stanford CoreNLP	Java NLP analysis tools	POS tagging, NER, dependency parsing, lemmatisation, coreference resolution
http://nlp.stanford.edu/software/corenlp.shtml		
VENSES	SWI-prolog semantic evaluation system for recognising textual entailment	Functional and syntactic constituency, topic identification (main, secondary, potential topics)
http://project.cgm.unive.it/venses_en.html		
WordNet	Lexical database	Lexical generalisation
http://wordnet.princeton.edu/		
VerbNet	Class-based verb lexicon	Predicate generalisation
http://verbs.colorado.edu/m̃palmer/projects/verbnet.html		
SENNA	ANSI C NLP predictions	POS tagging, chunking, NER, semantic role labelling, syntactic parsing
http://ml.nec-labs.com/senna/		

METEOR	Machine translation evaluation system	Segment alignment (based on exact word, stem, synonym and paraphrase)
http://www.cs.cmu.edu/ alavie/METEOR/		

For modelling language

Lemur	Text search and ranking	Information retrieval, text mining
http://www.lemurproject.org/		
KenLM	C Library for Language modeling	idem
http://kheafield.com/code/kenlm/		
SRILM	C Library and executable for Language modelling	idem
http://www.speech.sri.com/projects/srilm/		

For machine learning

Weka	Java software for machine learning	Machine learning, data mining
http://www.cs.waikato.ac.nz/ml/weka/		
SVM-rank	C Implementation of the Support Vector Machine algorithm	Binary/ multiple rankings based on features provided
http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html		

SVM-tree kernels	idem	Measures similarity between syntactic sub-tree structures
http://disi.unitn.it/moschitti/Tree-Kernel.htm		

Table E.2: List of software tools and resources used throughout the study

E.3 Plagiarism Case Studies

16th May 2012

The Romanian education and research minister stepped down following investigation that found substantial amount of plagiarism in his papers. http://www.nzz.ch/aktuell/international/ioan-mang_1.16913287.html

30th March 2012

The Hungarian president had his doctorate revoked as large parts of plagiarism were found in his thesis. http://www.euractiv.com/central-europe/hungarian-president-loses-doctorate-plagiarism-case-news-511869

5th March 2011

News report revealed the numbers of reported plagiarism and other academic misconduct cases across the UK. http://www.telegraph.co.uk/education/educationnews/8363345/The-cheating-epidemic-at-Britains-universities.html

A full list of incidents reported is available online

http://www.telegraph.co.uk/education/educationnews/8363783/University-cheating-league-table.html

1st March 2011

The German defence minister had to step down after his doctoral thesis was accused of plagiarism. His thesis was investigated by online collaboration and it was found that many parts were word-for-word plagiarism, and the source texts were highlighted in various colours.

http://www.tagesschau.de/inland/guttenberg762.html

31st December 2011

A new method to detect plagiarism is to set up an online platform for collaborative manual detection. This has contributed to the zu Guttenberg case mentioned below. http://de.vroniplag.wikia.com/wiki/Home

7th August 2011

It is always a difficult task to determine who the original author was. The question is, what if two authors published the same materials and there is an argument of not knowing who came first?

http://www.spiegel.de/unispiegel/studium/professor-contra-doktorandin-wer-klaut-hier-bei-wem-a-776909.html

13th July 2011

The first plagiarism scandal in Germany has brought up more similar cases. The German representative to the EU has been accused of improper referencing thus has his doctorate rescinded.

http://www3.uni-bonn.de/Pressemitteilungen/198-2011

17th May 2010

An alleged plagiarism case on a published book had begun an online battle between the accuser and the accused.

http://www.zeit.de/studium/hochschule/2010-05/mathematik-plagiate

17th June 2011

It took over a year to finally put a ban on publishing the plagiarised book.

http://www.mathematik.uni-marburg.de/g̃umm/Plagiarism/index.htm

12th November 2010

In this article, a ghost writer tells his story. He works for paper mills and has written about 5,000 pages of literature.

http://chronicle.com/article/The-Shadow-Scholar/125329/