Authorship attribution of forum posts

Femke Klaver
Huygensstraat 42
1813XJ Alkmaar
Tel.: 06-18176240
Student number: 1895745
E-mail address: f.klaver@student.vu.nl

Faculty of Arts
rMA Linguistics
Supervisor: Prof. Dr. P.T.J.M. Vossen
Second reader: Dr. H.D. van der Vliet

Media and Network Services
Media Mining
Supervisor: Drs. M.M. Spitters

October 2014

I hereby declare that this dissertation is on original work, written by myself alone.
Any information and ideas from other sources are acknowledged fully in the text and notes

Alkmaar, 21 November 2014 Femke Klaver

Acknowledgments

With this thesis I will finalize the Research Master in Linguistics. When I started at the VU over 5 years ago, I could not have imagined that I would end up specializing in computational linguistics. Looking back, I cannot imagine that I would not. Developing programming and machine learning skills and applying those skills for linguistic research is, as far as I am concerned, the best of both worlds.

I would like to take this opportunity to thank a number of people. First of all I want to express my gratitude to Piek Vossen. Not only because he supervised this thesis, but also for creating the opportunity to specialize in computational linguistics at the VU. When Chantal van Son and I asked for the possibilities of specializing in computational linguistics, he and Hennie van der Vliet came up with a linguistic engineering program, for which I am very grateful.

This thesis would not have been possible without the Media & Network Services department at TNO. It was a great experience to conduct my research and write my thesis from an internship. I want to thank everybody at TNO who supported me in any way. Martijn Spitters has been a great supervisor, giving me loads (and loads) of constructive criticism. But I especially want to thank him for always having faith in my competence. I also want to thank Mark van Staalduinen, for always being positive, energetic and full of ideas.

Finally I would like to thank my friends and family for supporting me and keeping me motivated.

Abstract

The anonymity of dark webs such as Tor facilitates a relatively safe environment for criminal activity. Because criminals tend to leave as few traces as possible, identification of those users often is difficult. One trace they do leave, however, is their writing style. The aim of this research is to investigate the discriminative possibilities of certain writing style characteristics, or stylometric features, when applied to messages posted on a forum on Tor.

Two different problems regarding authorship analysis are investigated. The first problem entails identifying users that use multiple aliases. We look at the differences between two sets of texts to see whether or not they are written by the same person. Our approach shows very promising results when character bi- and trigrams are used as features. These features return a precision score of .98 and a recall score of .9.

The second problem focuses on attributing the correct author to a text. Experiments are carried out using instances containing feature values of single messages and average feature values of combinations of messages as input data. In this thesis we show that, when we use combinations of 5 messages, we are able to select the correct user from a set of 67 users with an accuracy of around 83%.

Both approaches deliver very promising results, showing that writing style characteristics are indeed very helpful in discriminating between users. While this research focused solely on forum messages, the same approaches can be applied on data from other sources. They can also be helpful for linking data from different sources to each other.

Contents

Chapter 1

Introduction

This thesis investigates the possibilities of de-anonymizing individuals on the *dark web* based on their writing style. With more and more web-sites storing IP-addresses of visitors on the one hand, and placing tracking cookies on your computer on the other, the internet cannot ascertain the anonymity of its users. Therefore, some users tend to reside to the *dark web*, that tries to keep its users as anonymous as possible. The Tor network, one of the most popular examples of a *darknet*, was originally created to provide a platform for freedom of speech. Currently, however, this network also provides a place where criminals can reside anonymously. On so-called *hidden services*, which often include anonymous discussion forums, they can, for example, sell and buy drugs or even hire a hitman.

Due to the enhanced anonymity, the law enformance is having difficulties locating criminals residing on the dark web. The dark web enables criminals to leave as little traces as possible. One trace they do leave, and which could help identifying individuals, is their writing style. The research in this thesis focuses on using writing style, or *stylometric features*, to decide which author most probably wrote a certain text. Therefore it can be placed within the field of authorship analysis, the field that is interested in gathering information about authors of texts.

Much research that focuses on authorship analysis considers only a small number of authors and usually with relatively much data per author available. The seminal work by Mosteller and Wallace (1963), for example, focuses on the authorship analysis of only two authors. Moreover, Mendenhall (1887), believes that at least 100,000 words per author are necessary to create a 'characteristic curve' for an author. The number of words that is believed to be necessary decreased in the century after Mendenhall. (Ledger and Merriam, 1994), for example, argue that authorship characteristics are only apparent in text samples containing more than 500 words. However, the vast majority of messages on underground forums contain less than 500 words making it more difficult for authorship analysis.

With the rise of internet and social media, more and more researchers became interested in authorship attribution for a larger number of possible authors, with a smaller number of words per author available (e.g. Argamon et al., 2003; Zheng et al., 2006; Luyckx and Daelemans, 2008; Abbasi and Chen, 2008). Research has focussed on a lot of different types of digital communication, including e-mails (De Vel, 2000), tweets (Layton et al., 2010) and newsgroup messages (Zheng et al., 2006). But not only the small amount of available data and large number of possible authors are problematic. Because some users operate under multiple aliases, you cannot always be certain that two user names belong to two different

users. Those users can cause confusion when you try to attribute an author to a text, as there is an overlap in writing style (Johansson et al., 2013; Novak et al., 2004).

Although recent work shows promising results for authorship attribution of short texts, many results are obtained using a controlled data-set considering messages on a single topic. Novak et al. (2004), for example, show that when they include texts about different topics, their results decrease significantly compared to using messages from a single topic. As filtering data by topic reduces the amount of data, this is not desirable for real-life situations with only a limited amount of data available.

1.1 Problem definition and hypotheses

The aim of this thesis is to find an approach that is able to attribute authors to messages on anonymous web forums. The resulting approach should have as less interference from topical characteristics as possible. This leads to the main question of this thesis:

> *To what extent can we identify individuals by their typical writing style on anonymous web forums?*

The intent is not to find a real name behind a text, but to select the correct author of a text from a set of multiple authors. Due to the large number of users on web forums, it is not expected that the system will be able to assign a text to just a single possible author. Instead, the hypothesis is that it is more likely that the resulting system is able to assign a text to a top-n of possible authors.

In order to answer the main question, two sub-questions are formulated:

1. Which feature sets and combinations of feature sets work well for authorship attribution of forum posts and which do not?
2. To what extent are we able to reduce the influence of topic?

To investigate which feature sets work well, we use the framework for authorship attribution proposed by Zheng et al. (2006) as a starting point. This framework consists of multiple feature sets, based on characters, words, function words, structure and content. Because of the promising results of character n-grams (Forsyth and Holmes, 1996; Stamatatos, 2008) and syntax (Baayen et al., 2002), these features are added to the framework. The aim is to select features that minimize the influence of topic. Certain features, such as features based on content, can influence the performance of the system. Content features, for example, may work well when an author writes about a single topic, but may cause confusion when an author writes about multiple topics. These features are tested on data obtained from a forum called Black Market Reloaded, a hidden service on the Tor network.

The research is split up into two different problems. For the first problem we aim to answer the question whether or not two sets of messages are written by the same user. For the second problem, we aim to attribute the correct author to an unseen text, based on previously seen messages. The different problems are introduced in more detail in sections 1.1.1 and 1.1.2.

1.1.1 One user with multiple aliases

The first problem focuses on identifying users that operate under multiple aliases. In order to solve this problem, two different experiments are carried out. For the first experiment,

an artificial environment is created by splitting up one user into two pseudo-users, because we do not know which users have multiple aliases. To identify users with multiple aliases, we compare the results from a clustering algorithm using input from two pseudo-users to the results from two different users. The hypothesis is that the clustering algorithm does not perform very well when applied to the pseudo-users, as they are actually just one user. The scores should be better when the algorithm is applied to two different users. Significantly lower scores for the two pseudo-users than for the two different users, can be used as an indication that two user names belong to the same user.

For the second experiment, messages from users are split up into two sets. The variance between the average feature values of sets of the same user and between sets of different users are used as input for a classification algorithm. Intuitively, the variance between sets of messages from different users is larger than between sets of messages from the same user. Therefore, the hypothesis is that it can give an indication whether two sets of messages are written by the same or by different users.

1.1.2 User ranking for unseen posts

The second problem focuses on selecting a top-n of possible users for an unseen post, based on previous posts. A classifier is trained using one part of a user's messages, and then tested using the remaining part. For each test instance, the system decides how often the correct user is in the top 1, 3 or 5 of most possible users. Aiming to reduce the influence of topic and anomalous messages, such as short thank you notes, experiments are also conducted using the average feature values of 3 and 5 messages as input.

To see whether or not this approach is also suitable for other languages, these experiments are repeated for Dutch messages.

1.2 Thesis outline

The remainder of this thesis is structured as follows. Chapter 2 gives an overview of previous research relevant for this thesis. It focuses authorship attribution in general, authorship attribution of short texts, and identifying users with multiple aliases. Chapter 3 first gives a quantitative analysis of the data and how the data is prepared. Secondly, it gives an overview of the feature extraction process, the different approaches that are employed and how these approaches are evaluated. The results of these approaches are given in chapter 4 and discussed and interpreted in chapter 5. This thesis finalizes by giving the main conclusions in chapter 6.

Chapter 2

Background and Related work

This chapter is divided into two main parts. The first part of this chapter provides a background on authorship attribution. More specifically, it aims to give a better understanding of what the field of authorship attribution entails and the different techniques that are used. The second part shows how these techniques are used in previous work on authorship attribution. It first focuses on authorship attribution from a broader perspective in paragraph 2.2. Many articles, however, consider authorship attribution for long texts, such as literary works. Because the research described in this thesis uses short and informal texts as dataset, it secondly focuses on authorship attribution specifically for informal texts in paragraph 2.3. Another aim of this thesis is to investigate the possibilities for identifying users that operate under multiple aliases. Therefore, paragraph 2.4 discusses previous work on the subject of anti-aliasing.

2.1 Background

2.1.1 Stylometry

Authorship attribution is usually based on *stylometric features* for the identification of an author. As the name already suggests, the basis for those features lays in a field called *stylometry*. Holmes (1998) describes stylometry as follows:

> At its heart lies an assumption that authors have an unconscious aspect to their style, an aspect which cannot consciously be manipulated but which possesses features which are quantifiable and which may be distinctive. (Holmes, 1998, pp. 111)

The part of the writing style of an author that is formed unconsciously forms the basis for authorship attribution. This means that, for example, content words that can easily be manipulated by the author do not denote writing style as much as, for example, function words.

One of the most influential works in stylometry is the seminal work by Mosteller and Wallace (1963), who show the distinctive capabilities of high-frequent function words. In the decades after Mosteller and Wallace, however, it was assumed that lexical features were the most effective in capturing writing style. Tallentire (1973), for example, claimed that 'no potential parameter of style below or above that of the word is equally effective in establishig objective comparison between authors and their common linguistic heritage'. Three decades

later Holmes (1994) stated something similar, namely that '[...] yet, to date, no stylometrist has managed to establish a methodology which is better able to capture the style of a text than that based on lexical items'.

The remainder of this paragraph gives a brief overview of the different types of features that are used for authorship attribution. It does not aim to provide a thorough insight into all possible features and implementations, but it aims to show the basic concepts. How the different features are implemented in related work is discussed throughout 2.2 and 2.3. The feature implementation for this research will be given in paragraph 3.2.1.

For the description of the different features I mainly follow Zheng et al. (2006) and Abbasi and Chen (2008). While they do not include all possible features that are traditionally used for authorship attribution, the features they do provide form a good baseline. Abbasi and Chen (2008) distinguish two main categories: *static* and *dynamic* features. Static features are 'well-defined context-free categories' (Abbasi and Chen, 2008, pp. 11), that is, the features are defined regardless of the contents of the message. Dynamic features on the other hand, only exist when they appear in the data. To illustrate, when n-grams are used as feature, the instances of that feature consist of the n-grams that appear in the data, rather than a fixed set of words. This has the consequence that n-grams that do not appear in the data are not included as features. The framework provided by Zheng et al. (2006) only contains static features. Abbasi and Chen (2008) build upon that framework, but extend it by adding dynamic features.

Static features include the following types:

1. *Lexical features*
   Lexical features can be divided into two sub-categories: *character* and *word* features.

   (a) *Character features*
   Features based on characters include the number of characters, single upper- and lowercase characters, but also special characters, spaces and digits.

   (b) *Word features*
   Features based on words include the number of words used in a text, the number of short words, average sentence length, but also the number of characters that appear in words. Measures for lexical richness, such as the number of words that occur only once (hapax legomena) or twice (hapax dislegomena) are also examples of word features.

2. *Syntactic features*
   Throughout the literature, there are different opinions about which features should be labelled as syntactic. According to Zheng et al. (2006), syntactic features consists of the counts of punctuation marks and function words.[1] Counts of part-of-speech tags may also be included (Abbasi and Chen, 2008).

3. *Structural features*
   Features based on structure do not relate to the contents of a text, but to the presentation of a text. Features include whether a text contains a greeting or quoted content, but also the number of paragraphs.

[1] Hirst and Feiguina (2007) argue that these features are actually *lexical* features. Although they make a fair point, we will follow the definition by Zheng et al. (2006), because our research builds upon their framework.

4. *Content features*
   Content features include a predefined set of words specific for the contents of a text. An example of a content-specific feature for a text discussing trade would be "sell" or "price". The predefined set of content-specific words for, for example, legal texts would contain different words.

Dynamic features include several n-gram frequencies. N-grams capture combinations of characters, words or part-of-speech tags, consisting of n elements. For example, character bigrams ($n = 2$) for "for example" would be:[2] |fo|, |or|, |r_|, |_e|, |ex|, |xa|, |am|, |mp|, |pl|, |le|. Word bigrams for the first sentence of this paragraph would be: |Dynamic features|, |features include|, |include several|, |several n-gram|, |n-gram frequencies|.

Abbasi and Chen (2008) use the following dynamic features:

1. *Lexical features*
   - Character bigrams (e.g. aa, ab)
   - Character trigrams (e.g. aaa, aab)
2. *Syntactic features*
   - Part-of-speech tag bigrams (e.g. NN, VB)
   - Part-of-speech tag trigrams (e.g. NN, VB, DT)
3. *Content features*
   - Word unigrams
   - Word bigrams
   - Word trigrams

2.2 Authorship attribution in general

One of the first attempts to authorship attribution is found in Mendenhall (1887). He uses word length distribution to create what he calls a 'characteristic curve'. He claims that his method is indeed capable of creating such curves, but that at least 100,000 words are necessary. When less words are used, he expects that personal characteristics are overshadowed by accidental variations.

A real breakthrough in authorship attribution came with the seminal work by Mosteller and Wallace (1963), who try to attribute authorship to the disputed Federalist papers. Because the truth about which author wrote which paper is not known, Mosteller and Wallace can only say something about the probability that an author wrote a text. To obtain these probabilities, they use predefined sets of words, including mostly function words, but also some content words. They show that high-frequent function words work surprisingly well as features to discriminate between authors.

After Mosteller and Wallace (1963), Burrows (1989) also showed the potential of function words. He shows that high-frequent function words are very capable of distinguishing between different authors. Diederich et al. (2000) not only show the promising results of using function

[2] "|" denotes the n-gram boundary, and "_" a single space character

words as features for authorship attribution, they are also one of the first to show the potential of using *support vector machines* (SVM[3] for authorship attribution. When applied on German newspaper articles, SVM outperforms different techniques, such as Naive Bayes, kNN and Decision trees.

Baayen et al. (1996) explore the usefulness of syntax for authorship attribution and how that compares to an approach using lexical features. To capture the syntax of sentences they use rewrite rules based on the outcome of syntactic parsing. The occurences of those rewrite rules in a text are then used to distinguish between two authors. Their research suggests that syntax-based methods leads to better results than methods based on lexical features.

Koppel et al. (2009) test how different features perform on different text types. They use three different corpora for their investigation: one set of e-mail messages between two authors, a corpus of books from the 19th- and early 20th- century, and a set of blog posts by 20 bloggers. They found that for all three corpora the best results were obtained by a combination of content words and character bigrams, which both had the highest information gain in the training corpus. They do note, however, that these features "might [...] produce overly optimistic results that will not be borne out in real-life applications" (Koppel et al., 2009, 12).

2.3 Authorship attribution for short texts

Luyckx and Daelemans (2008) point out that, due to the small number of possible authors and the large amount of available data, previous research on authorship attribution often uses 'unrealistic sizes of training data', resulting in overestimated results. In realistic situations, such as the analysis of student essays or forum posts, the available data often contains less words than the data that is normally used in studies considering authorship attribution. However, using a small number of words for authorship attribution is often claimed to be infeasible. In 1887, Mendenhall claimed that the enormous amount of 'one hundred thousand words will be necessary and sufficient'. More than a century later, the necessary number of words has decreased, e.g.:

> We do not think it likely that authorship characteristics would be strongly apparent at levels below say 500 words, or approximately 2500 letters. Even using 500 word samples we should anticipate a great deal of unevenness, and that expectation is confirmed by these results. (Ledger and Merriam, 1994)

Although this is, of course, a significant decrease, the majority of forum messages contains less than 500 words. And the messages are not only short, the text size is also highly dependent on the function of the message: A message that aims to thank someone is mostly very short, while a message that aims to explain something can be very long.

More and more studies investigate the possibilities for authorship attribution in more realistic situations. De Vel (2000), for example, investigates the possibilities for e-mail messages. His corpus contains a total of 274 documents from five authors. With features based on words, characters, syntax and structure and an SVM classifier, his study results in 71-84% accuracy. De Vel et al. (2001) also focus on e-mail messages. They do take another aspect in consideration: the influence of topic. Their corpus contains a total of 156 documents from 3 authors. The documents are divided into three categories: *movies*, *food* and *travel*. They

[3]for more information about SVM, see section 3.2.2.

show that this division causes a decrease in performance as opposed to lumping all documents together, most probably due to the decreased number of documents per author.

Zheng et al. (2006) provide a framework for authorship attribution of forum posts. Rather than using a small number of authors as in the more traditional works on authorship attribution, they experiment using sets of 5, 10, 15 and 20 authors. They also experimented with different subsets of messages, ranging from 10 to 30 messages per author. Using lexical, syntactic, structural and content features, they report an accuracy of 97.69%. Without structural features and content specific features, the accuracy drops to 90%.

Still, the accuracy obtained by Zheng et al. (2006) seems very promising. Another problem, however, is that the time span of the data is not clear. The data is extracted from a Usenet newsgroup of which the first messages date back to 1994 and the last messages are from May 2014.[4] Zheng et al. (2006) do not specify whether they used messages within a specific time span, or that they used the whole range. The fact that the digital writing conventions changed considerably since the 1990's may attribute to the high accuracy scores acquired by their studies.

Building upon the framework by Zheng et al. (2006), Abbasi and Chen (2006) developed a method they call *Writeprints*. Rather than using a feature set that is similar for all authors, this method entails constructing a *Writeprint* for each author based on the author's key features. Features that are important for one author but not for another are used as pattern disruptor to decrease the level of stylistic similarity between two authors. Finally, they compare the *Writeprint* of a text to the *Writeprint* of each author to attribute the right author to a text. The *Writeprints* method is tested on four different corpora: Enron e-mail, Ebay comments, Java Forum and CyberWatch Chat. For each corpus, they selected 100 authors. The chat-corpus contained the least words per author: 1,422. The other corpora contained significantly more words per author: 23,423 words for the Ebay comments, 27,774 for the Enron e-mail and 43,562 words per author for the Java Forum.[5] The *Writeprints* methods resulted in an accuracy of 31.7% for the chat-corpus, 52.7% for the Java Forum, 91.3% for the eBay comments and 83.1% for the e-mail corpus. When including less authors, accuracy scores improved (Abbasi and Chen, 2006).

Hirst and Feiguina (2007) use bigrams of syntactic labels as features to distinguish between writings from Charlotte and Anne Bronte. Their books are not particularly short, so for the experiments they split the books into smaller blocks of 200, 500 or 1,000 words, depending on the experiment. Although Hirst and Feiguina (2007) focus on using short texts, Luyckx and Daelemans (2008) argue that, due to the high number of texts, the size of their training data is still unrealistic. Also, while they indeed focus on short texts, the data is still obtained from literary works.

As opposed to the literary works used by Hirst and Feiguina (2007), Layton et al. (2010) aim to attribute authorship for Twitter messages that contain at most 140 characters. They show that using n-grams as features results in an accuracy of over 70%. *@replies* in the messages are an important factor in achieving that result. When removed, the accuracy drops with 27% to 43%.

While promising results are emerging for handling the small amount of words per author, an additional difficulty is that the data is often imbalanced. That is, the number of posts differs greatly between the authors of forum posts. Stamatatos (2008) refers to this as the

[4] For an example of a thread, see https://groups.google.com/forum/#!forum/misc.forsale.computers.memory

[5] The texts extracted from the Java Forum did contain programming code instead of natural language.

class imbalance problem and investigates the possibilities to reduce the negative effect of imbalanced data. He proposes an approach that creates many short text samples for authors with a low number of posts and less, but longer, text samples for authors with a high number of posts. This method is able to increase the accuracy for the minority authors, with only a slight loss of accuracy for the majority authors.

2.4 Anti-aliasing

A different problem that arises when applying authorship attribution techniques to forum posts, is that people can have multiple user accounts. This is closely related to the field of authorship analysis that considers *similarity detection*. There are a number of reasons why a person would feel the need to create multiple aliases. Johansson et al. (2013) introduce the following:

1. the old alias has been deleted due to inactivity
2. the old password has been forgotten
3. the old alias has been banned by a moderator
4. the old alias has lost the trust of other members
5. bad relationships have been developed with other members
6. the user wants an extra alias to support own arguments, cause debate or controversy (the extra alias used for purposes of deception is sometimes referred to as a sockpuppet)
7. the user wants to discuss immoral or illegal activities
8. the user wants anonymity due to privacy reasons

For the first two reasons, the user would probably not attempt to disguise the fact that he created multiple aliases. For reasons 3-6, on the other hand, it would be more problematic if it was found out. Users who have multiple aliases for those reasons might alter their writing style in order to reduce the chance of getting caught. Afroz et al. (2012) address that topic, and aim to detect deception in writing style. Using stylometric features, they try to identify authors who intentionally obfuscate their writing style, but also authors that try to imitate the writing style of another author. They show that it is more difficult to identify authors who obfuscate their own writing style than authors that imitate another author's writing style.

Novak et al. (2004) use an unsupervised clustering algorithm in order to detect users with multiple aliases. They do not take potential obfuscation of writing style into account, but focus solely on the identification of users with multiple aliases. They created an artificial environment by splitting the data from each user into two pseudo-users. Using data from 100 users as input resulted in an accuracy score of around 80% with 50 messages per user. Increasing the number of messages per user to 125 resulted in an accuracy of around 95%. These results, however, are obtained by using messages considering a single topic.

Abbasi and Chen (2006) do not only focus on authorship attribution, but also on similarity detection. Rather than using an instance-level approach where each message is used separately to represent a user, they use an identity-level approach where all messages from a user are lumped together. Similar to Novak et al. (2004), the data from each user was first

split up to create two pseudo-users. Using data from 100 different users, their *Writeprints* method resulted in an F-measure of 85.56% for the E-mail data set, and 94.59% for the eBay comments. For the data obtained from the Java forum and CyberWatch Chat, the F-measure was lower, with 76.87% and 49.91% respectively. These results are in line with the results for their authorship attribution task.

Johansson et al. (2013) do not only employ stylometric features in order to identify users with multiple aliases, but also use the time of day as an indicator. They split each user in their corpus into two pseudo-users, resulting in user set A and user set B. To test whether or not two users are actually the same user, the similarity between time-profiles and stylometric features of each user in set A and each user in set B is calculated. These scores are then used to rank how similar the users of set B are to the users of set A. Their results suggest that combining time-profiles with stylometric features delivers the best results, with an accuracy of over 70% for 50 users. When the number of users is increased to 250, the accuracy drops to 55%.

Bu et al. (2013) try to detect sockpuppets in Wikipedia using stylometric features. They found that a number of features also included in Zheng et al. (2006) were useful for sockpuppet detection. They also found that two additional features, more specific for online communication, were helpful for similarity detection: beginning of a sentence without capital letter, and no white space between sentences. With their system, they were able to detect sockpuppets with an accuracy score of 68%.

2.5 Summary

In this section we showed several approaches for authorship attribution and similarity detection, which are the two main tasks addressed in this thesis. However, some of the approaches introduced above fit our specific tasks better than others. As shown, many works deal with a small number of authors and with a large number of texts per author available. For this research it is the opposite: it deals with a large number of authors (67 for English and 25 for Dutch), but with a relatively small number of texts. Not only the number of texts is small, the texts are often relatively short.

The work by Zheng et al. (2006) specifically considers authorship attribution of forum posts. Their framework does not only yield promising results for their own studies, but also when it is used as a basis in other works (Abbasi and Chen, 2006; Bu et al., 2013). Besides the framework of Zheng et al. (2006), character n-grams perform very well for multiple authorship attribution tasks considering short texts (Abbasi and Chen, 2006; Hirst and Feiguina, 2007; Layton et al., 2010). Therefore, it is worth investigating how well the features from the framework by Zheng et al. (2006) and character n-grams perform when applied to our data set.

A problem that arises specifically for the similarity detection task is that, with anonymous forum data, you cannot be sure which aliases belong to the same user. Abbasi and Chen (2006) and Johansson et al. (2013) show a solution for this problem, by splitting one user into two pseudo-users. This enables you to test how well the approach performs in distinguishing between texts that are written by the same user and those written by different users.

Chapter 3

Data and Methods

3.1 Data

This section provides detailed information about the data used for this project. It starts by shortly describing from where the data was collected. Secondly, it describes how Dutch and English messages were extracted. In the third part it gives a quantitative analysis. Finally it describes how the data is further prepared for the experiments.

3.1.1 Data description

The data used for this research is extracted from a forum on the Tor network: Black Market Reloaded (BMR).

BMR mainly concerns the trade in illegal goods, such as drugs, weapons and stolen credit cards. Messages do not only concern trading goods, they also include, for example, feedback of buyers and sellers, identification of possible scammers and tips about operating anonymously. BMR is a multilingual forum and contains messages in English, Dutch, Spanish, German, among others. In total 23 languages were detected using a language detector.

The messages from the BMR forum are collected in the period from October 2012 to January 2014. For each topic, all posts are extracted and for each posts the user id, user name, topic id, name of the topic, time of the post, post id, and the contents of the post are stored in a database. The database contains a total of 92,333 posts from 8,348 users. The messages are posted in 12,923 different topics.

3.1.2 Data selection

Because this research focuses solely on messages written in English or Dutch, the data first has to be filtered for language. Language detection was done automatically. The language detector uses a list of frequent words and character trigrams to decide on the language, resulting in 66,491 English messages and 3,340 Dutch messages.

Some forum messages proved problematic for the classifier and were classified incorrectly. These errors are caused by sloppy grammar and spelling, the use of abbreviations and loanwords, for example:

(1) *Mocht ik in deze business verder gaan dan zal ik je zeker pm'en! Jack*
Would I continue in this business, I will definitely send you a pm! Jack

(2) *De sample is erg plakkerig! Veel THC dus :-D. Ik heb erg veel vertouwen in Drugs-Mania ;-)*
The sample is real sticky! So there's much THC :-D. I have high confidence in Drugs-Mania ;-)

The examples above are clearly written in Dutch and spelling is not sloppy in these cases. However, example 1 got classified as German and example 2 as Polish.

Because only a limited amount of data is available for Dutch, useful information could be lost by these mistakes. To obtain as much Dutch messages as possible, an extra step is included for the selection of the data. First of all, only the messages of individuals that posted at least 10 times were extracted. This filtered out almost all the messages that were incorrectly classified as being either English or Dutch. For English, this resulted in 49,870 messages and for Dutch 2,142.

While the number of English messages could be considered as being large enough for this research, the number of Dutch messages is limited. Because a number of Dutch messages are incorrectly classified as being another language, these messages are initially not selected. Therefore the following workaround was used: for all individuals that posted a Dutch message more than 10 times, all the messages that are not classified as either English or Dutch are extracted. With this workaround, another 2,027 messages were retrieved. These were manually annotated for language. This resulted in another 647 English messages and 223 Dutch messages. For the English messages, however, this resulted in users with less than 10 English posts. Their messages are removed from the data.

In total, 50,498 English messages and 2,333 Dutch messages were selected. The next paragraphs describe the frequency distributions of those posts.

3.1.3 Frequency distribution of forum posts

English

The selection procedure described in paragraph 3.1.2 resulted in 1,148 users posting in English, throughout 8,774 different topics. Table 3.1 gives an overview of the frequencies of the English messages. It shows that in total, there are 50,498 english messages. The minimum number of messages by a user is 11, because users with less messages are filtered out. The most-frequent posting user has posted 1,865 messages, while the average is only 43. The messages contain over 3,5 million tokens.[1]. The user with the least tokens only posted 144 tokens and the user with the most posted almost 83,000. On average, users posted 3,095 tokens.

	Total	Min	Max	Mean
Posts	50,498	11	1,865	43
Tokens	3,553,806	144	82,784	3,095

Table 3.1: Frequencies of English posts and tokens

Looking at the top 10 of users, as illustrated in figure 3.1, we can see an explanation for the relative low mean. While the top user posted 1,865 messages, the fifth user posted only

[1] A token represents the number of elements a message contains after tokenization. This can be either a word, punctuation mark, symbol or emoticon

half of that, with 993 messages. This number is even lower for the tenth user with the most messages, who posted only 384 messages.

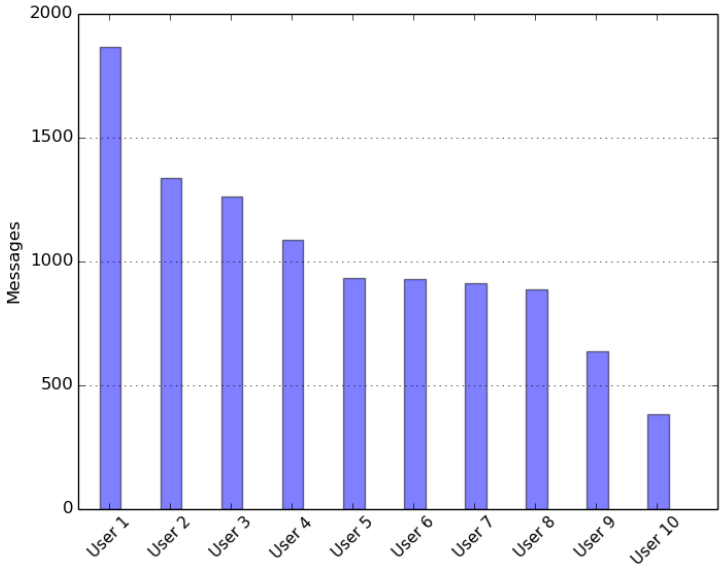


Figure 3.1: Overview of the top 10 English users

A small investigation of the further distribution of the posts pointed out that 93% of the users posted less than 100 messages and more than half of the users posted less than 25 messages on the forum. Figure 3.2 shows how the forum posts of users with less than 250 messages are distributed.

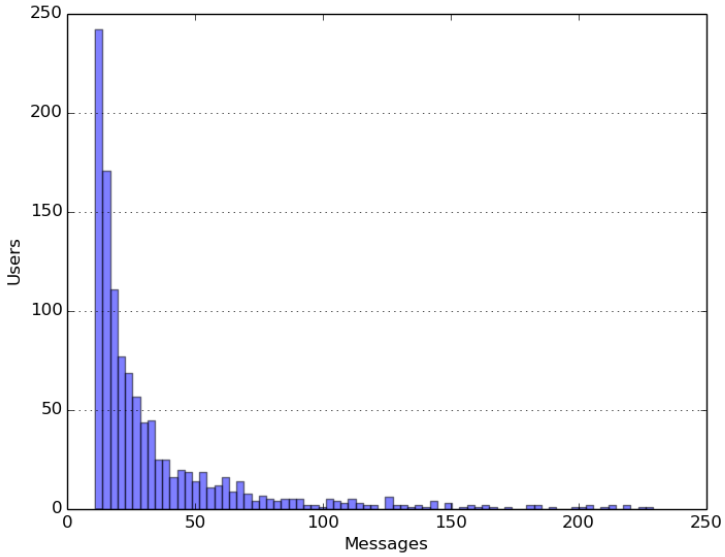


Figure 3.2: Distribution of forum posts for English users with less than 250 messages

Dutch

For Dutch, the overall frequencies are considerably lower than for English. Rather than the 1,148 English users, there were only 67 Dutch users. As shown in table 3.2, there are 2,333 messages posted in Dutch. Again, the user with the least posts has posted 11 messages because users with a lower number of messages are filtered out. The highest number of messages posted by a user is 295, considerably less than the 1,865 messages of the most-frequent posting English user. On average, however, Dutch users posted 43 messages, which is the same as for English. Although the average number of messages is similar, the average number of tokens per user is not. While for English the average number of tokens is 3,095, for Dutch this is only 1,375. The user with the least tokens has only posted 42, and the user with the most tokens has posted 16,037.

	Total	Min	Max	Mean
Posts	2,333	11	295	34
Tokens	92,186	42	16,037	1,375

Table 3.2: Frequencies of Dutch posts and tokens

The top 10 Dutch users represented by figure 3.3 show something similar as the top 10 English users. The top user has posted a relatively large number of messages: 295. In this case, the post frequency of the second user is already more than twice as low as the post frequency of the first user. The fifth user only posted 85 messages, less than a third of the messages of the first user.

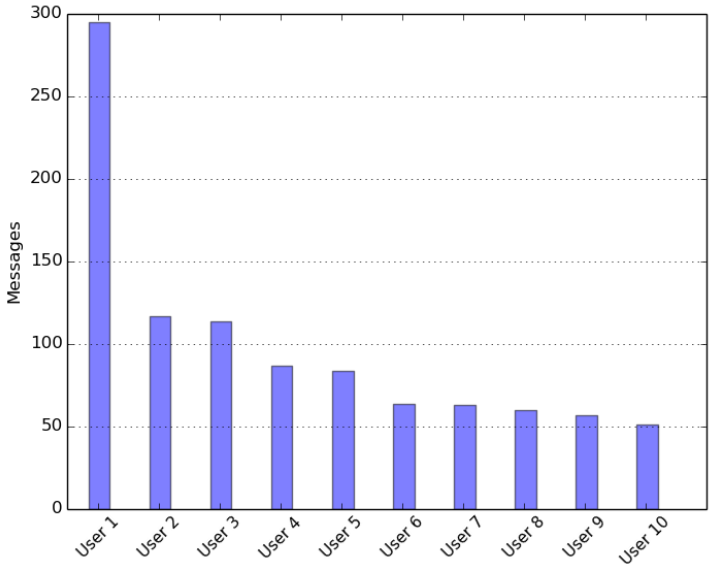


Figure 3.3: Overview of the top 10 Dutch users

Figure 3.4 shows the distribution of Dutch forum messages. It shows that most users posted less than 50 messages and only a three users posted more than 100 messages.

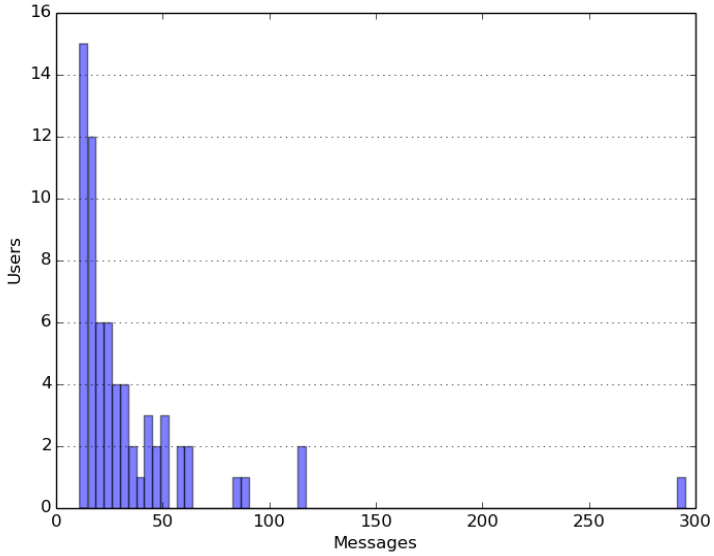


Figure 3.4: Distribution of Dutch forum messages

3.1.4 Data preparation

Data extracted from internet forums needs some preparation before it is possible to extract features. This section describes how the data was prepared for the feature-selection process, and the problems that arose during those preparations.

The data preparation roughly consists of three steps:

1. Clean data
    - Remove emoticons
    - Extract hyperlinks, remove from string and put in separate field
    - Add spaces after interpunction
2. Parse
    - Tokenize
    - Lemmatize
    - Tag for parts-of-speech
3. Export to JSON

The data cleaning is done using regular expressions in Python. The Dutch texts are parsed using Frog (Van den Bosch et al., 2007) and the English texts are parsed using Pattern (De Smedt and Daelemans, 2012). Using those packages, the texts are tokenized, lemmatized, tagged for parts of speech and parsed for syntactic function. Frog and Pattern use different tag sets (Frog uses the tags developed for the Spoken Dutch Corpus ("Corpus Gesproken Nederlands", or CGN (Van Eynde, 2001)) and Pattern uses the tags developed for the Penn Treebank (Santorini, 1990)). Using two different tag sets is not ideal, as it would make the

feature-extraction process too language dependent. Because the Penn tag set is used more generally, for example, the Stanford part of speech tagger (Toutanova et al., 2003) also uses it, the CGN part of speech tags are converted to the Penn tags. The exact mappings can be found in Appendix A.

After cleaning, tokenizing, lemmatizing, tagging and parsing, the pre-processed data are formatted as JSON and written to a file. Below is an example to illustrate the JSON structure.


```
[
  {
    "Hyperlinks": [],
    "Post": {
      "PostID": "100315",
      "TimeStamp": "2013-12-01 23:44:00",
      "TopicID": "14251",
      "TopicName": "RIP Sheep market"
    },
    "Sentences": [
      [
        {
          "lemma": "ja",
          "rel": "B-TSW",
          "tag": "UH",
          "token": "ja"
        },
        {
          "lemma": "man",
          "rel": "B-NP",
          "tag": "FW",
          "token": "man"
        },
        {
          "lemma": "alla",
          "rel": "B-VP",
          "tag": "FW",
          "token": "alla"
        },
        {
          "lemma": "akbar",
          "rel": "B-NP",
          "tag": "FW",
          "token": "akbar"
        }
      ]
    ],
    "String": "ja man alla akbar",
    "User": {
      "UserID": "1535020",
      "UserName": "smokesomeweed"
    }
  }
]
```

Each entry in the structure not only contains information about the words in the message, but also information about the time of the message, the topic the message was posted in and whether or not the message contains hyperlink(s).

3.2 Methods

This section describes the methods that are used for the two different problems introduced in section 1: the first problem focuses on the identification of users with multiple user names and the second problem focuses on attributing the correct author to a forum message.

Based on the promising results of using machine learning techniques for authorship attribution (e.g. Diederich et al., 2000; Novak et al., 2004; Zheng et al., 2006; Abbasi and Chen, 2006, amongst others), machine learning techniques are used in order to solve the problems described above. For the first problem both a clustering as well as a classification approach is implemented. For the second problem, only a classifying approach is used.

The section is structured as follows. First it describes the feature selection procedure, which is the basis for each experiment. Secondly it gives some insight into the machine learning algorithms that are used. Sections 3.2.3 and 3.2.4 deal with the actual experiments. Each section describes the data selection, tools and the experimental setup. The experiments for the first problem are carried out using only English forum posts, while the experiments for the second problem are first carried out using English forum posts and are repeated using Dutch forum posts. All machine learning tasks in this project are carried out using the implementations of the algorithm in the Scikit-learn package for Python (Pedregosa et al., 2011).

3.2.1 Feature selection

The features used for the experiments are based on the framework by Zheng et al. (2006), as already discussed in section 2. Some of the features, however, cannot be used for this research. Besides once- and twice-occurring words, vocabulary richness measures are excluded. A number of structural features are excluded as well, because the database does not contain all the necessary information. For example, information about paragraphs is not stored, making it impossible to extract the paragraph-related features.

The feature set proposed by Zheng et al. (2006) is also enriched with a number of features. Abbasi and Chen (2008) showed that adding character bigrams and trigrams enhance the results of the framework by Zheng et al. (2006), but the good performance of character n-grams is also shown in research (e.g. Cavnar et al., 1994; Kešelj et al., 2003; Peng et al., 2003). Therefore, those features are added to the framework. Because Baayen et al. (1996) show the potential of using syntactic information for authorship attribution, part-of-speech tag trigrams are also added to the framework. Solorio et al. (2011) show the potential of using the omission of spaces as feature for similarity detection. A preliminary empirical investigation of the data showed there are indeed some users that do not use white spaces around punctuation, while other users do. Therefore, the omission of spaces is added to the framework. Table 3.3 shows the complete list of features used for the experiments.

Feature	**Normalized by**
Lexical features	
- *Character-based features*	
Total number of characters	-
Total number of alphabetic characters	C
Total number of upper-case characters	C
Total number of digit characters	C
Total number of white-space characters	C
Total number of tab spaces	C
Frequency of letters (26 features)	
Frequency of special characters (21 features)	C
Character bigrams	N
Character trigrams	N
- *Word-based features*	
Total number of words	-
Total number of short words (less than four characters)	W
Total number of characters in words	C
Average word length	-
Average sentence length in terms of character	-
Average sentence length in terms of word	-
Total different words	W
Hapax legomena (Frequency of once-occurring words)	W
Hapax dislegomena (Frequency of twice-occurring words)	W
Syntactic features	
Frequency of punctuations	C
Frequency of function words	W
Structural Features	
Total number of sentences	-
Has a greeting	-
Has signature	-
Content-specic Features	
Frequency of content-specic keywords	W
Additional Features	
POS trigrams	N
No white spaces	-

Table 3.3: List of features and their normalization (C = number of characters in the message, W = number of words, N = number of n-grams. - means that the raw counts are used.)

As table 3.3 shows, most of the features are normalized. This means that the raw feature value is divided by the total number of characters, words or n-grams in a message, depending on the type of feature. The implementation of most of the features is relatively straightforward, because there are no different ways of, for example, counting the number of alphabetic characters.

For some features, however, the implementation is not so straightforward. One of those features is the *frequency of function words*. Zheng et al. (2006) extract function words using a list of 150 words and assign each function word as a different feature. García and Martin (2006), on the other hand, extract function words using part of speech tags and use the function word/content word ratio as one feature. The research presented in this thesis uses the same approach as Zheng et al. (2006), because this does not depend on a part of speech tagger. This means that for English, the same function words as in Zheng et al. (2006) are used. For Dutch, such a list could not be found. In order to construct a list of Dutch function words, the *Algemene Nederlandse Spraakkunst* (Haeseryn et al., 1997) is consulted.[2] This resulted in a list of 150 function words, which are included in appendix B.

Not only function words, but also the content specific keywords can be implemented in several ways. Zheng et al. (2006) use a list of 11 intuitively chosen keywords. Another way of choosing keywords, is to use a filter that returns words that are specific for a context in comparison to another context. For this research, the words in the corpus of forum messages are compared to the words in a large background corpus of news messages, using a approach related to *tf-idf* (Salton and Buckley, 1988). This approach retrieves the prior of a word in the background corpus. Only words with a low prior are selected as content-specific words, as they do not occur often in the background corpus.

3.2.2 Machine Learning techniques

For the experiments, two different types of machine learning techniques are used: clustering and classification. Clustering is an unsupervised technique, meaning that it uses the data without predefined labels as input. Classification, on the other hand, is a supervised technique where labeled data is used to train a classifier, which is then tested by assigning labels to unseen cases. Both techniques are explained more thoroughly in the next paragraphs. While the first problem employs both techniques, the second problem focuses solely on classification. This section will not give a fully detailed explanation of the calculations behind the techniques, but it does explain the basic concepts of the algorithms used in the experiments.

Clustering

One clustering algorithm that is often used for many different problems and is also used for an experiment in this thesis is k-means. In short, k-means first computes k random cluster centers and aims to minimize the distance between each data point (in this case the extracted features for each message) and the center it is assigned to. Therefore it iterates over the data points and recomputes the cluster centers at each iteration until it stabilizes, that is, the

[2] Auxiliaries: http://ans.ruhosting.nl/e-ans/02/02/02/body.html
Articles: http://ans.ruhosting.nl/e-ans/04/body.html
Pronouns: http://ans.ruhosting.nl/e-ans/05/body.html
Prepositions: http://ans.ruhosting.nl/e-ans/09/03/02/body.html
Conjunctions: http://ans.ruhosting.nl/e-ans/10/02/body.html for co-ordinating and
http://ans.ruhosting.nl/e-ans/10/03/body.html for sub-ordinating conjunctions.

cluster centers do not change position significantly in the following iterations, meaning that the minimum distance between the centers and the different data points is reached (for more information, see Arthur and Vassilvitskii, 2007).

A downside of k-means is that you have to specify the number of clusters, or k. Consequently, you have to know beforehand by how many authors the messages you want to cluster are written. Although it is not known whether two user names actually belong to one or two authors, k-means can still be used as a method. The idea is that if k is set to 2 and two user names belong to one author, k-means will not be able to create two clearly defined clusters, but that if two user names belong to two authors, it will be.

Classification

Based on promising results in previous work (e.g. De Vel, 2000; De Vel et al., 2001; Zheng et al., 2006), the classification experiment is carried out using an SVM (support vector machines) classifier.[3]

The idea of SVM is that it searches for a line, or *plane*, between two or more classes. As shown by figure 3.5, such a plane can be made in different ways.

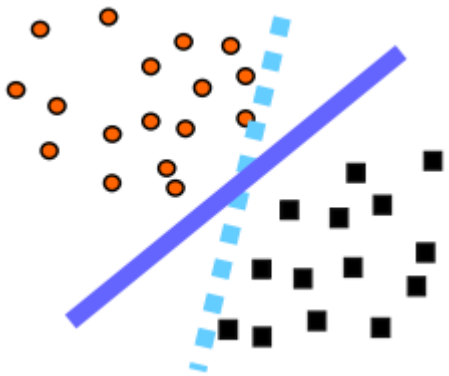


Figure 3.5: Two possible linear discriminant planes (Bennett and Campbell, 2000)

The aim of SVM, however, is to search for the optimal plane, or *hyperplane*, that maximizes the distance between positive and negative cases. Figure 3.6 shows the hyperplane that has the maximum margin between the two classes. This is established by "pushing" the different possible planes until they reach a small number of data points, or *support vectors*, for each class. The plane with the biggest margin is the hyperplane that is used to classify new data points.

SVM can use different kernel functions, of which two are used in the experiments below: a *Linear* and *RBF* kernel. The difference between these kernels is that the linear kernel only allows for a linear hyperplane, while the RBF kernel also allows for a non-linear hyperplane. A linear kernel is often preferred for high-dimensional data, with a large number of classes and features (Fan et al., 2008; Chang and Lin, 2011).

The first problem is a binary classification task. The second problem, however, entails a multi-class classification task, with 67 classes for the English data set and 25 for the Dutch data set. In this case, each class represents a user. To deal with multiple classes, there are

[3]Small preliminary experiments were conducted using both SVM and Naive Bayes as classifier. Because SVM returned the same or better scores, Naives Bayes is not used for any further experiments.

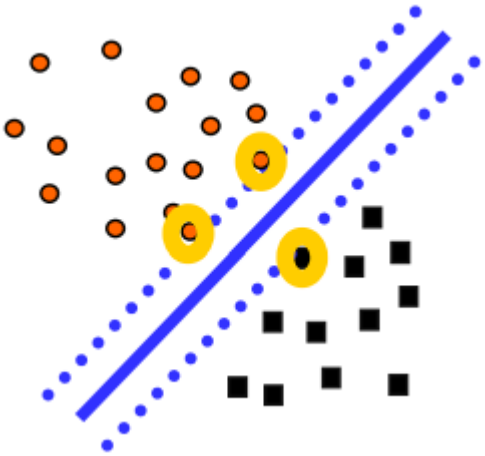


Figure 3.6: Best hyperplane with support vectors (Bennett and Campbell, 2000)

two options: the first option is generally referred to as a *one-vs-one* classifier, and trains multiple binary classifiers by comparing each class to all other classes separately. The other option, generally referred to as a *one-vs-rest* classifier, that creates multiple binary classifier by comparing each class to all other classes as a whole. For large and high-dimensional data-sets, the *one-vs-rest* approach is preferred (Fan et al., 2008; Chang and Lin, 2011).

3.2.3 Problem 1: One user with multiple aliases

The first problem considers the identification of users that use multiple aliases. Two different approaches are explored in order to solve this problem. The first approach incorporates a clustering algorithm to see how useful the different feature sets are for clustering messages from authors. The second approach tries to solve the problem using the same feature sets, but this time with a classification algorithm. The outcome of this approach is whether or not two sets of messages are written by the same author.

Select suitable users

To select the users for both approaches, we need to be sure that there is actually only one person behind a user name. Because the users from our data sets are anonymous, we do not know whether or not one user has multiple user names. The following steps decrease the risk of selecting a user name that is being used by multiple users, but also to decrease the risk that the selected set contains multiple user names that are operated by the same user:

1. Select users who posted between 75 and 200 messages.
2. Create time-profiles based on hours
3. Select users that have a time-profile with a block of sleep hours in which they post significantly less than during other hours. The block size is set at 7 hours.
4. Calculate distance between time-profiles of remaining users

The first step is applied to select users based on their number of posts. For this problem, the number of posts is set between 75 and 200. While this is a big range, it does not only filter out users with only a small number of posts, it also filters out users that have a relatively high post count because they usually have another function on the forum, for example forum admins and moderators. Secondly, a time-profile is created for each person in the group. As Johansson et al. (2013) point out, time-profiles can be really useful to identify users with multiple user names. A time-profile contains the total number of posts per hour, for example:

```
<24, ... , 0, 1, 4, 3, 2, 5, 3, 20, 11, 15, 25, 16, 12, 22, ... , 17>
```

These numbers depend highly on the total number of posts by the user. To obtain a time-profile independent of the total number of posts, the number of posts per hour are divided with the maximum number of posts in an hour, resulting in:

```
<0.69, ... , 0.0, 0.03, 0.11, 0.09, 0.06, 0.14, 0.09, 0.57, 0.31, 0.43, 0.71,
0.46, 0.34, 0.63, ... , 0.49>
```

The third step is to select users that have a block of sleep hours in which they post significantly less than during other hours. If users do not have a block of sleep hours, it could actually be the case that there are multiple users behind one user name. Therefore, they are filtered out. Also, the distance between time-profiles can be an indication that two user names are in fact used by two different users. Johansson et al. (2013) point out that when the distance between time-profiles is large, the possibility that two user names are being used by the same user is small and visa versa. Therefore, in our experiments users are only combined with the users with the greatest difference between time-profiles. This, however, does have the side-effect of selecting users based on the time-zones they live in. Although this is something to keep in mind, for this problem it is less problematic than the chance of combining messages of the same user. Making combinations of users is described more thoroughly in the next paragraphs.

Approach 1: clustering

The purpose of this approach is the see how the different features perform when clustering messages from different users. The hypothesis is that with messages from two users as input, the clustering algorithm is able to assign the messages from each user to a different cluster. To see if the results are similar for different sets of users, the experiment is repeated three times using different user sets. For each set, one user is randomly selected from the users that meet the sleep-block criterium and have posted between 75 and 200 messages. On the one hand, the post count is used to ensure that there is enough data per user, and on the other hand to filter users that have posted a relative large number of messages. These users are combined with the user that also meets these criteria and whose time profile differs the most from that of the first user. The differences between time profiles are calculated using the Euclidian distance metric.

Define baseline

The baseline is defined using character features and k-means[4] as the clustering algorithm. The goal of the clustering experiment is to see if a clustering algorithm is able to reconstruct

[4]more specifically, k-means++ (Arthur and Vassilvitskii, 2007).

the users as good as possible. Because it initially aims to cluster the data from two users, k is set to 2.

Experiments

The clustering experiment is repeated for all different feature sets, that is: character features, word features, syntactic features, structural features, content specific features, character n-grams, POS trigrams and the omission of white spaces. Because the results of these experiments were not very promising, with a highest V-measure of 0,184 for word features, no further clustering experiments are conducted. [5]

Evaluation

The performance of the different feature sets is evaluated using scores for homogeneity, completeness and the v-measure (Rosenberg and Hirschberg, 2007). Homogeneity measures whether the data points in a cluster all belong to the same class. If all data points indeed belong the the same class, the homogeneity score is 1. Completeness measures if all the instances belonging to a class are assigned to the same clusters. If the clusters are complete, that is, include all the instances of a class, the score is 1. If the clustering algorithm performs well on a task, clusters should be both homogeneous and complete. The V-measure represents a combination of homogeneity and completeness and is calculated by using the harmonic mean of these measures. Scatter plots are created to get a visual representation of the results and are included in Appendix D.1.

Approach 2: classification

The aim of this approach is to classify whether two sets of messages belong to the same user or to different users. Rather than using feature vectors of the messages as input for the classifier, this approach uses feature vectors of the differences between the average feature values of sets of messages. The idea behind this, is that it smoothens the data and reduces the potential influence of message function (e.g. offering goods or a thank you note) or message topic (e.g. drugs, weapons or credit card fraud). The remainder of this paragraph describes how these feature vectors are created and how the experiments are carried out and evaluated.

Creating feature vectors

Figure 3.7 shows a schematic overview of the different steps to extract a feature vector which is used as input for the machine learning algorithm. Per selected user, the messages are first randomly divided into two sets. For each message in a set, the features are extracted. Then, for each set the average feature values are calculated and used as a new feature vector. This means that per user, two feature vectors are created (e.g. A_1 and A_2).

The next step is to compare these different feature vectors and calculate the differences between the average feature values of each set. The final feature vector contains the value differences and is used as training instance for the machine learning algorithm. If the instance is computed using input from the same user (A_1 and A_2), the instance gets the label "same" and if it is computed using input from different users (A_1 and B_1), the instance gets the label "different". The resulting instance matrix are shown in figure 3.8.

[5] see section 4.1.1 for all results.

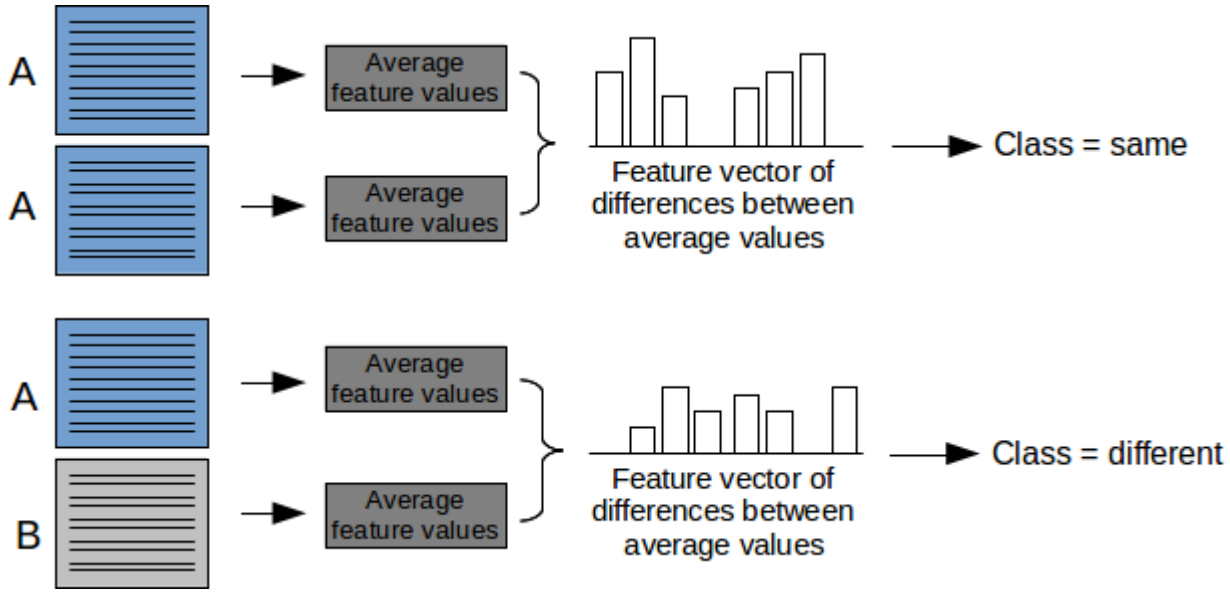


Figure 3.7: Machine learning instance creation

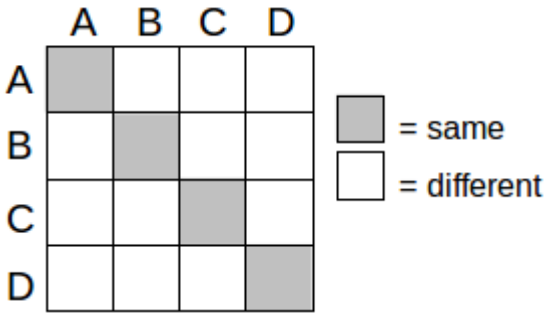


Figure 3.8: Combination of message sets with the class labels

This method has the effect that the dataset becomes more and more imbalanced, because the number of instances increases exponentially when more users are included. That is, with 4 users the ratio of positive/negative examples is 1/3 (4 positive examples and 12 negative examples). With 50 users, however, the ratio would be 1/49. To make the data more balanced the ratio is fixed at 1/5. For each user, 5 users with the least similar time profiles are selected to create the negative examples. This also decreases the risk of combining two user names that actually belong to the same user and label it as being from different user names.

The final step in the feature vector creation process is to normalize the feature values. The normalization process is done using the feature values for each instance, that is, for each feature the independent values are scaled depending on the highest and lowest values in the set of instances. Because some values are relative (and consequently below 1) and some values are raw counts (and consequently above 1), classifiers may create a bias towards the raw counts. Normalizing feature values to fit within a range of 0 to 1 prevents this.

Define baseline

Before conducting experiments with different features, a baseline needs to be defined. Following Zheng et al. (2006), the baseline is set using only character features as features and SVM with a linear kernel as classifier. To see whether the results depend on the way the data is divided, the experiment is repeated 5 times. The standard deviation is used to measure the variation between runs.

Experiments with different feature sets

To get an idea of which features work well to decide whether texts are written by the same user or not, the experiment is carried out using different feature sets. Zheng et al. (2006) use the following combinations of feature sets:

1. Character features
2. Character features + word features
3. Character features + word features + syntactic features
4. Character features + word features + syntactic features + structural features
5. Character features + word features + syntactic features + structural features + content-specific features

These combinations, however, are built upon the assumption that character features work well for authorship attribution, as they are used in all combinations, but it is not tested if character features are indeed the best performing features. Therefore, the first step is to test the performance of each feature set separately. This step is carried out using a SVM classifier with both a linear and an RBF kernel. Because the linear kernel outperformed the RBF kernel for almost all features, the remainder of the experiments are conducted using only the linear kernel. The second step is to combine the best performing feature set with all other individual feature sets, to see if a combination outperforms the individual feature set.

Evaluation

The experiments are evaluated using 5-fold cross-validation. This means that for each fold, 80% of the data is used to train the system and 20% is used to test the data. Each fold uses a different subset of data. The average recall, precision and F-scores of those folds are used to measure the performance of the system. Recall is used to measure how many of the available instances of a class are classified correctly, and is calculated using:

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{3.1}$$

Precision measures how many instances that are classified as a certain class actually belong to that class.

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{3.2}$$

As with the V-measure for clustering, that combines the homogeneity and completeness scores, the F-measure combines the recall and precision scores.

To see if the dividing and merging process influences the results, the experiments are repeated five times. If the division of the data of the users does influence the results, the performance of the classifier should return different results after each run. The final performance of the system is measured using the mean precision, recall and F-score of the 10 runs and the difference between the scores are measured using the standard deviation. Not only the overall results, but also the results per class are taken into account to see if the different feature sets and combinations perform differently for the two classes.

3.2.4 Problem 2: User ranking for unseen posts

Instead of looking ift two sets of texts are written by the same author, this problem focuses on attributing the correct author to a message. Due to the large number of authors (67) it is not expected that the system is able to select just one possible author for a text. Therefore, the hypothesis is that it is more likely that the correct author is in a top-n of most possible authors.

To address this problem, we do not only investigate how well the different feature sets perform using single messages as input, but we also look at how they perform using average feature values of combinations of messages as input. While the previous two experiments focused solely on English messages, this experiment is repeated using Dutch messages.

Select data

The English data is the same as the data used for the first problem in section 3.2.3. Initially, the Dutch data was selected using the same criteria as for the English messages, but this resulted in only 4 users. As a solution, the post count criterium that users should have posted between 75 and 200 messages is lowered to 25 and 100. This resulted in a data set containing 25 users. Lowering the post count criterium does introduce the chance that the results for the Dutch data set are less consistent, because there is not enough data available per user.

Feature vector creation

The experiments regarding the single messages use the same input as for the clustering experiment, as described in 3.2.3. The results for the first problem, described in section 4.1, indicate that using combinations of messages returns better results than using single messages. This means that the average feature values of multiple messages are probably a better representation for capturing the writing style of a user. To test if using combinations of messages increase the performance, feature vectors are created using the average feature values of three and five random messages. The final feature values are normalized to fit in a range from 0 to 1.

Define baseline

For this experiment, the baseline is set using character features and an SVM classifier with a linear kernel. While the first problem entailed a binary classification task, this problem entails a multi-class classification task. Because of the relatively large number of classes (67 for English and 25 for Dutch), a *one-vs-rest* approach is implemented, which is known to

be better at handling high-dimensional classification problems than the *one-vs-one* approach (Fan et al., 2008).[6]

As already mentioned, this experiment is two-fold: on the one hand it investigates the performance when single messages are used as input and on the other it investigates how using combinations of messages influences the performance. To see whether or not the different combinations of messages influences the performance, the experiment is repeated 5 five times. For each part, we do not only look at how often the correct user is classified, but also how often the correct author is in the top 3 or 5 of most possible authors.

Experiments with different feature sets

The experiments with different feature sets for this problem are similar to the experiments for the first problem, as discussed in paragraph 3.2.3. That is, first the best scoring feature set is defined and secondly it is tested whether or not combining that feature set with other feature sets improves the performance. What differs from the first problem, is that no experiments are conducted using different classifiers or, in the case of an SVM classifier, kernels. For this problem, only the one-vs-all SVM classifier with a linear kernel is used.

Evaluation

The performance of the system is evaluated using the accuracy score:

$$\text{Accuracy} = \frac{\text{Number of correctly classified messages}}{\text{True positives + true negatives + true negatives + false negatives}} \tag{3.3}$$

For the second part of the experiment, which aims to see how often the correct author is in the top 1, 3, or 5 most possible users, Scikit-learn's implementation of liblinear is slightly altered to obtain a list of all possible labels and their confidence scores. These confidence scores represent how sure the classifier is that a class label belongs to an instance. Per instance, all possible labels are ordered from high to low using the confidence score and the position of the correct label is stored. Finally, the percentage of instances where the correct label is in the top 1, top 3 or in the top 5 of most possible labels are calculated. These percentages are used as evaluation measure. As shown in equation 3.4, this measure is very similar to the accuracy score.

$$\text{Measure} = \frac{\text{Number of messages with the correct label in the top } n}{\text{Total number of messages}} \tag{3.4}$$

Whether or not a feature set significantly outperforms the baseline is tested with paired T-tests.

[6] see section 3.2.2 for more information about one-vs-one and one-vs-rest approaches.

Chapter 4

Results

This section presents the results of the two different problems investigated in this thesis. The first refers to the problem in which one user operates under multiple aliases. The aim is to answer the question whether or not different forum messages belong to the same user. The second problem refers to the user ranking for unseen posts and aims to assign the correct author to a forum message. Both problems are discussed more thoroughly in sections 3.2.3 and 3.2.4. The main outcome for both problems is that character n-grams are very suitable features to distinguish between texts of different authors. For the first problem, character bi- and trigrams return a precision score of .96 and a recall score of .84. Another outcome for both problems, but for the second problem in particular, is that using average feature values of a combination of multiple messages improves the performance of the classifier. Section 4.1 and section 4.2 show the results of the first and second problem respectively in more detail.

4.1 Problem 1: One user with multiple aliases

To deal with the first problem two different approaches are implemented. The first approach implements the unsupervised clustering algorithm k-means. The second approach uses the differences between average feature values of texts and the supervised classification algorithm SVM to decide whether a set of messages is written by the same of different authors. This section shows using a clustering algorithm is not a suitable approach for our task, but that the second approach using a classification algorithm works very well.

4.1.1 Approach 1: clustering with k-means

To investigate the possibilities of identifying a user with multiple aliases, a clustering algorithm is employed. The hypothesis is that if two aliases belong to a single user, the clustering algorithm is not able to create clearly defined clusters based on authorship, as opposed to when two aliases belong to two different users. Because for our data set it is not known whether or two aliases belong to the same user, the messages from one user are randomly split to create two pseudo-users, The results from using messages from one user are compared to the results from two different users. To test if the selected users and user combinations influence the performance, the experiments are repeated for three different users and user combinations. With none of the feature sets the clustering algorithm could create clear clusters based on authorship. The algorithm does return clusters, but these are most likely

based on topic or the function of a message. This indicates that an unsupervised method like k-means does not seem to be the appropriate approach for this task.

Baseline

Following Zheng et al. (2006), the baseline for all experiments is set using character features (e.g. number of character, number of upper- and lowercase characters, special characters and spaces). For this experiment, it is expected that using the messages from the same user as input does not result in clearly defined clusters based on authorship, as opposed to messages from two different users. Table 4.1 shows the clustering results when one user is used as input. Homogeneity, completeness and V-measures are scores between 0 and 1, where a score near 0 indicates that the clusters do not represent the ground truth and a score near 1 indicates that the clusters do represent the ground truth. Table 4.1 shows very low scores for all three measures.

Users	Homogeneity	Completeness	V-measure
User A_1 + A_2	0.003	0.003	0.003
User B_1 + B_2	0.02	0.024	0.021
User C_1 + C_2	0.029	0.067	0.04

Table 4.1: Clustering scores for messages from a single user as input.

The very low homogeneity scores for all three users mean that the data points in the clusters do not belong to the same class. The low completeness scores show that not nearly all instances of a certain class are assigned to the same cluster. The low V-measure combines the homogeneity and completeness scores and gives a more general indication that k-means does not perform well. This supports the hypothesis that when one user is split into two pseudo-users, the clustering algorithm will not return well-defined clusters.

However, as table 4.2 shows, the different scores are almost equally low when the messages of two different users are used as input, which again means that no well-defined clusters are created. Although the scores for the combination of user A and D and the combination of user B and E are slightly better than the scores for the single users, the scores for combination C and F are actually worse. This does not support the hypothesis that when messages from two different users are used as input, the clusters would be better defined than when messages from one user (two pseudo-users) are used as input.

Users	Homogeneity	Completeness	V-measure
User A+User D	0.003	0.022	0.005
User B+User E	0.075	0.092	0.083
User C+User F	0.002	0.055	0.004

Table 4.2: Clustering scores for messages from two different users as input.

The above suggests that using k-means together with character features cannot cluster the messages based on the users. More specifically, it does not enable us to define whether or not two user names belong to the same user.

Experiments with different feature sets

In the previous paragraph it is made clear that character features do not perform well in clustering messages from different authors. The aim of this paragraph is to see if other types of features outperform character features and can thus give a better indication whether or not two user names belong to the same user.

Featureset	User A_1+A_2	User A+D	User B_1+B_2	User B+E	User C_1+C_2	User C+F
CharFeatures	0.003	0.005	0.02	0.024	0.04	0.005
WordFeatures	0.014	0.184	0.013	0.005	0.0	0.002
SyntacticFeatures	0.029	0.015	0.002	0.027	0.013	0.007
StructuralFeatures	0.051	0.043	0.15	0.002	0.002	0.024
ContentSpecific	0.015	0.058	0.15	0.005	0.004	0.001
Char n-grams	0.012	0.006	0.59	0.005	0.02	0.063
POS n-grams	0.015	0.006	0.15	0.006	0.02	0.007

Table 4.3: V-measures of the feature sets for different users and user combinations.

Table 4.3 shows the V-measures of the clustering experiment for both the messages of a single user (split into two pseudo-users) and messages of two different users. It shows that the performance highly depends on the user or user combination. For example, using the messages from user A and D as input returns better results ($M = .184$) then using only the messages from the pseudo-users A_1 and A_2 ($M = .014$). On the other hand, if the messages of user B and E are used, the score for both users ($M = .005$) is lower than the score for using only the messages from the pseudo-users B_1 and B_2 ($M = .013$). Besides the inconsistent performance, the overall performance is very low. The highest obtained V-measure is .184, which indicates that the created clusters are not very well defined, but, as the second highest score is only .063, it is an exceptionally high score compared to the rest of the scores.

These results indicate that this approach is not suitable to identify multiple aliases that belong to the same user, as the results for two different users are not better than the results for two pseudo-users. The low scores for messages from different users indicate that the algorithm is not able to create clusters based on authorship. Because of the inconsistent performance and low scores, no further experiments with k-means were conducted.

4.1.2 Approach 2: classification

Because the results from the clustering experiments are not very promising, a alternative approach is implemented. The second approach differs from the first in two ways: rather than using a clustering algorithm, a classification algorithm is employed, and instead of feature values of single messages, feature values of differences between combinations of messages are used. The expectation is that a combination of messages gives a more general representation of a user's messages, ironing out extremely short messages or differences in topic.

To combine the messages, the messages of a user are first randomly split into two subsets and the average feature values of those sets are calculated. To obtain the feature vectors for classification, the subset from a user is compared to the other subset of the same user and the variances are used to create an instance of the label *same*, and to five subsets from other users to create instances of the label *different*. This way we expect the learning algorithm to learn the difference between message sets from the same user as opposed to message sets

from different users. The algorithm is tested using 5-fold cross-validation. This means that for each fold, 80% of the data is used to train the algorithm, and 20% is used to test the performance.

This section describes the results of this approach. First, section 4.1.2 shows the results for the different feature sets separately, leading to the conclusion that character n-grams perform the best with an F-score of .89. Secondly, section 4.1.2 shows that combining character n-grams with other feature sets does not result in significantly better scores.

Baseline

As with the previous approach, the baseline is set using character features. Character features return a precision score of .75 (SD = .04), a recall score of .83 (SD=.05) and an F-score of .78 (SD=.04). The precision score means represents the percentage of correct assignments of a class (e.g., 95% for class *same*), and the recall score represents the percentage of instances that is retrieved (e.g., 87% for class *same).* The F-score represents the harmonic mean of precision and recall.

Class	P (SD)	R (SD)	F (SD)
Different	0.95 (0.02)	0.87 (0.03)	0.91 (0.02)
Same	0.54 (0.06)	0.79 (0.10)	0.64 (0.06)
Average	0.75 (0.04)	0.83 (0.07)	0.78 (0.04)

Table 4.4: Precision, recall and F-scores and their variance for both classes and the average scores using character features.

The results are shown in more detail in table 4.4. It shows that the relatively low precision score of .75 is due to the instances based on messages from the same user. While the precision score for instances labeled as *different* return a precision score of .95, the precision score for instances labeled as *same* is only .54. This means that, in 46% of the cases, sets of messages that are actually written by different users get classified as written by the same user. The difference between the recall scores is smaller, but the variance for the *same* category is higher, indicating a less stable performance. That is, the way the messages are split influences the recall score more for instances created using messages from the same users than for instances created using messages from different users.

Single feature sets

This section describes how well the other feature sets perform in deciding whether two sets of messages are written by the same or by different users, compared to the baseline. Because the aim is to find the best performing feature set, it only focusses on feature sets that outperform the baseline. Results for all feature sets are included in appendix D.2.

Figure 4.1 shows the performance of the different feature sets. It shows that content specific and syntactic features and character n-grams outperform the baseline. Paired samples t-tests were conducted to test whether or not these differences in performance are significant. There is a highly significant difference between the F-scores of character features (M=.78, SD=.04) and content specific features (M=.85, SD=.04), t(9)=7.96, $p < .001$. The F-score for syntactic features (M=.87, SD=.04) is also significantly higher, t(9)=10.75, $p < .001$. Character n-grams return the highest F-score (M=.89, SD=.03) which is highly significant

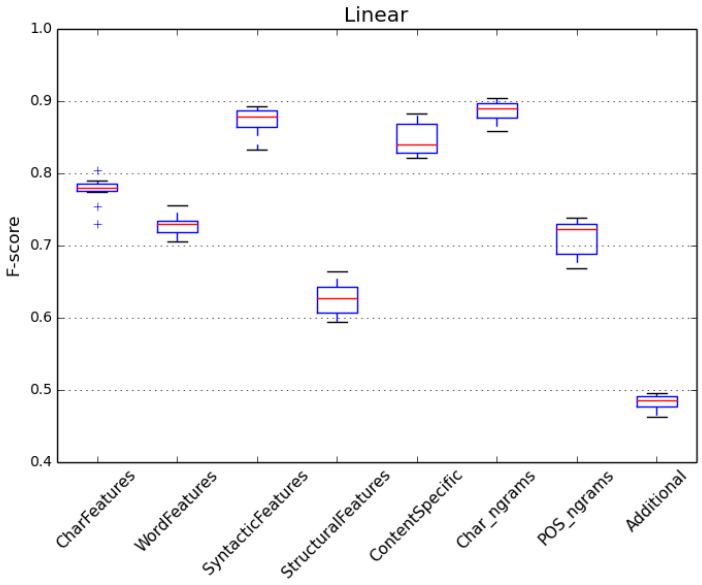


Figure 4.1: Box plot of the scores for all feature sets. The height of the boxes represent the variance between the scores of different runs. The horizontal lines in the boxes represent the average scores.

as well, t(9)=14.08, $p < .001$. The differences in F-scores are mainly due to the differences in precision scores, which are all highly significant, and to less extent to differences in recall scores. While the differences in recall between character features (M=.83, SD=.05), syntactic features (M=.86, SD=.04) and content specific features (M=.86, SD) are significant (syntactic features: t(9)=2.40, $p < .05$, content specific features: t(9)=2.68, $p < .05$), the recall score of character n-grams is not (M=.84, SD=.03), t(9)=0.88, $p = 0.4$.

Looking more closely at the scores shown in table 4.5, it shows that, although the overall scores of character n-grams are higher than character features, the recall score for the instances labeled as *same* is lower than the recall score of the baseline for the same category. Despite this lower score, the performance is more stable throughout the different runs (SD=.06) than the performance of character features (SD=.1).

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.01)	0.97 (0.01)
Same	0.98 (0.04)	0.68 (0.06)	0.80 (0.05)
Average	0.96 (0.03)	0.84 (0.03)	0.89 (0.03)

Table 4.5: Precision, recall and F-scores and their variance for both classes and the average scores using character n-grams

Although the recall score is lacking, an F-score of .89 indicates that character n-grams perform very well in deciding if two sets of texts are written by the same or different users. The fact that other feature sets return higher recall scores raises the question if a combining the overall well-performing character n-grams with another feature set can improve the recall score.

The experiment was repeated using an RBF kernel. For all feature sets, a linear kernel returned better results than an RBF kernel. Although the performance of the RBF kernel often improves after parameter optimization, the optimal parameters differed greatly per run making it difficult to actually improve the performance of the classifier. Therefore, the remainder of the experiments is carried out using a linear kernel.

Combinations

This section describes the results of the experiment that is carried out to see if combining character n-grams with another feature set improves the performance. As discussed in the previous paragraph, character n-grams return precision scores for both categories and the recall for the *different* category of 1, but the recall score is relatively low with a score of .68. Because for other categories the recall score is higher, it is interesting to see whether combining n-grams with another feature set is able to improve the recall score.

Table 4.6 shows the overall results for the combinations of feature sets. It shows that none of the combinations returns significantly higher scores than when only character n-grams are used.

Features	P	(SD)	R	(SD)	F	(SD)
Char n-grams	0.96	(0.01)	0.83	(0.02)	0.88	(0.02)
Ngrams+Char	0.97	(0.01)	0.83	(0.04)	0.88	(0.03)
Ngrams+Word	0.97	(0.01)	0.84	(0.02)	0.89	(0.02)
Ngrams+Synt	0.96	(0.02)	0.85	(0.03)	0.89	(0.02)
Ngrams+Stru	0.97	(0.01)	0.83	(0.02)	0.88	(0.02)
Ngrams+Cont	0.97	(0.02)	0.85	(0.03)	0.89	(0.03)
Ngrams+POS_	0.95**	(0.01)	0.81**	(0.02)	0.86***	(0.02)
Ngrams+Addi	0.96	(0.01)	0.82	(0.03)	0.87	(0.02)

Table 4.6: Average precision, recall and F-scores and their variance for combinations of character n-grams and other feature sets.

A more thorough investigation of the scores shows that none of the combinations returns higher recall scores for the category based on messages from the same user. Only the combination of n-grams with POS trigrams had a significant effect on the performance, but rather than increasing it, it decreases it. Tables of the scores per category are included in appendix D.2.2.

4.1.3 Preliminary conclusion problem 1

In the section above we discussed the results for the first problem, which investigates the performance of different feature sets for identifying users with multiple aliases. For this problem we employed two different approaches: the first approach tries to cluster single messages with k-means, while the second approach tries to classify subsets of messages with an SVM classifier. The expected outcome for the clustering approach was that using input from two pseudo-users - representing one user with multiple aliases - would yield lower scores then using input from two actually different users. The approach did not return very satisfactory results for any of the feature sets. The scores for different users were very low, and sometimes even lower than the scores for pseudo-users. This indicates that the clusters are not created

based on authorship and that this is not the appropriate approach for our problem. It is more likely that the clusters are created based on the topic of the message.

For the classification approach we used differences between subsets of messages to classify whether or not two subsets are written by the same or by different users. One subset from a user is compared to the other subset of the same user to create an instance of the label *same*, and to five subsets from other users to create instances of the label *different*. This approach did return very promising results. Especially character n-grams perform very well in distinguishing between subsets of messages written by the same and by different users, returning an F-score of .89. This suggests that the variance between subsets of messages indicate whether or not those two subsets are written by the same user. As opposed to the clustering approach, the topic of the messages does not seem to have much influence. One explanation is that using combinations of messages, rather than single messages, reduces the influence of topic.

4.2 Problem 2: User ranking for unseen posts

Rather than focusing on whether texts are written by the same or by different authors, the second problem focuses on attributing the right user to a forum post. This is done using a classification algorithm, which means that one part of the messages is used to train the classifier, and the remaining part - unseen by the classifier - is used to test the performance of the trained classifier. Because combining multiple messages into one instance returned promising results for the first problem, as described in section 4.1.2, experiments with combinations of messages are also carried out for this problem. This section shows that using combinations of messages indeed increases the performance.

To get an indication whether character n-gram also return the best results when applied to Dutch data, or that different feature sets deliver a better performance, the experiments are also carried out using Dutch data. The results in section 4.2.2 show that, for Dutch, character n-grams indeed deliver the best performance as well. In addition, it shows that using combinations of 5 messages improves the results, but also introduces more variation between different evaluation runs.

4.2.1 English

This section discusses the results for the experiments carried out using the English dataset. The main outcome of the experiments is that when combinations of 5 messages are used as input, the accuracy increases to .81, compared to an accuracy of .49 for single messages. Using single messages, the correct user is in the top 5 of most possible users in 71% of the cases. When combinations of 5 messages are used, this percentage increases to 95%.

Single messages

The results of the experiments conducted using single messages are shown in this section. The main finding is that character n-grams outperform the baseline as well as all other feature sets. Using character n-grams, the classifier selects the correct user from 67 different users for 49% of the test instances. 71% of the time, the correct user is in the top 5 of most possible users.

Baseline

Similar to the first problem, the baseline for this task is set using character features. Table 4.7 shows the results.

	Mean	(SD)
Top 1	0.11	(0.16)
Top 3	0.23	(0.18)
Top 5	0.32	(0.18)

Table 4.7: Percentages and their variance scores of cases where the correct user is in the top 1, 3 or 5 of most possible users, using character features.

It shows that in only 11% of the cases the classifier was able to select the correct author for a forum post from 67 different authors. The standard deviation of .16, which is even

higher than the mean score, indicates that there is much variation between the scores per user. For some users (almost) no messages could be assigned correctly, while for other users the percentage of messages that are assigned correctly is higher. However, for this problem the hypothesis was that the system would not necessarily be able to assign the correct user to a text, but that a system would be able to assign a text to a top-n of most possible users. Table 4.7 shows that the correct user is in the top 3 of most possible users in 23% of the cases, and in the top 5 in 32% of the cases. With a standard deviation of .18, the variation between users is even slightly larger than the variation for the top 1.

Experiments with different feature sets

This section gives an overview of the performance of the different feature sets, and whether or not they are able to outperform the baseline. The performance of the different feature sets is shown in figure 4.2.

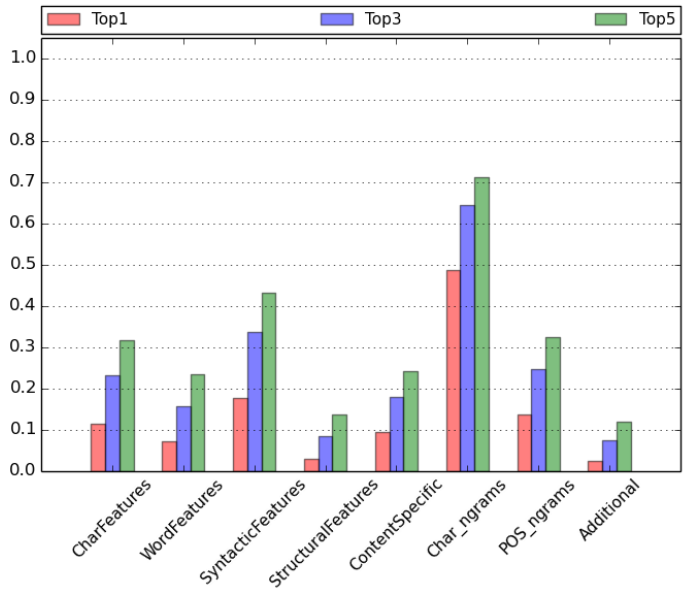


Figure 4.2: Ranking results for the English data set using single messages as input

It shows that from the different feature sets, only syntactic features and character n-grams deliver a better performance than the baseline. Syntactic features do not improve the performance as much as character n-grams, but the improvement is highly significant for all three categories. For the authors selected by the classifier, or top 1 the performance improves from M=0.11 (SD=0.16) to M=0.18 (SD=0.13), t(770)=3.95, $p < .001$. Although the correct author is only selected for 18% of the messages, the correct author is in the top 3 of most possible authors more often (M=0.34, SD=0.15), t(770)=5.23, $p < .001$, and even more often in the top 5 (M=0.43, SD=0.14), t(770)=5.85, $p < .001$.

The difference in performance between character features and character n-grams is larger, with an accuracy score of .49 (SD=0.18), and is highly significant, t(770)=16.71, $p < .001$. For 65% of the messages, the correct author is in the top 3 of most possible authors (SD=0.17), t(770)=17.38, $p < .001$. Another highly significant difference, t(770)=17.23, $p < .001$, is found for the top 5 of most possible users, that includes the correct author for 71% of the

messages (SD=0.15). These results support the hypothesis that it is more likely that the correct author is in a top-n of possible authors, rather than actually selecting the correct author.

The next step was to combine the best scoring features, character n-grams, to see if this could increase the system's performance.

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
Char n-grams	0.50	(0.18)	0.66	(0.16)	0.72	(0.14)
Ngrams+Char	0.50	(0.18)	0.66	(0.16)	0.72	(0.14)
Ngrams+Word	0.50	(0.18)	0.66	(0.15)	0.73	(0.13)
Ngrams+Synt	0.50	(0.18)	0.66	(0.16)	0.73	(0.14)
Ngrams+Stru	0.50	(0.18)	0.66	(0.16)	0.72	(0.14)
Ngrams+Cont	0.50	(0.18)	0.66	(0.16)	0.73	(0.14)
Ngrams+POS_	0.47***	(0.19)	0.64***	(0.17)	0.72	(0.15)
Ngrams+Addi	0.50	(0.18)	0.66	(0.16)	0.72	(0.14)

Table 4.8: Results for combinations of character n-grams with other feature sets using single message as input, *** = $p < .001$.

Table 4.8 shows that, similar to the results of the first problem, combining character n-grams with other feature sets does not lead to an improvement of the system's performance. Part-of-speech n-grams even decrease the performance significantly for the top 1 (M=0.47, SD=0.19); t(770)=4.78, $p < .001$ and top 3 (M=0.64, SD=0.17); t(770)=3.57, $p < .001$. For the top 5, however, there is no significant difference in performance between character n-grams (M=0.66, SD=0.16) and character n-grams combined with POS n-grams (M=0.72, SD=0.15); t(770)=1.28, $p = 0.2050$.

Combinations of messages

Although the aims and methods for the first problem (as described in section 3.2.3) are not easily comparable to the aims and methods of this problem, one of the main findings was that constructing training and test instances from several messages rather than from single messages returns very promising results, as shown in section 4.1. Therefore, the question arises if using combinations of messages leads to an improvement of the classifier. A combination of messages gives a more average representation of the author's messages than just a single message. For example, anomalous messages, such as thank you notes, are ironed out when they are combined with other messages to create a feature vector of the average feature values. Combinations of messages are created by calculating the average feature values of n-messages. This section shows the results for the experiments using average feature values of three and five messages.

Again, using character n-grams yields the best performance. Table 4.9 shows the average results for the average feature values for 1, 3 and 5 messages using character n-grams as features. To test whether the combining messages influences the performance of the system, the experiment is repeated 5 times. In this case, the standard deviation does not represent the variation between scores per author, but the variation between runs. A higher standard deviation means that different combinations of messages influence the system, indicating that its performance is less stable.

	Top1		Top3		Top5	
	Mean	(SD)	Mean	(SD)	Mean	(SD)
1 message	0.50	(0.01)	0.66	(0.01)	0.72	(0.01)
3 messages	0.70	(0.02)	0.85	(0.02)	0.89	(0.02)
5 messages	0.81	(0.02)	0.92	(0.01)	0.95	(0.01)

Table 4.9: Percentages of cases where the correct user is in the top 1, 3 or 5 of most possible user, using single messages and combinations of 3 and 5 messages. Character n-grams are used as features.

Table 4.9 shows how using combinations of messages influences the results. It shows that using the average feature values of three messages the accuracy increases to .70, as opposed to .50 for single messages. Using average feature values of five messages, the accuracy increases even further to .81. Similar results are found for the top 3 and 5. In all cases, the standard deviation is not very high, indicating that it does not matter which specific messages are combined. Figure 4.3 gives a visual representation of the results.

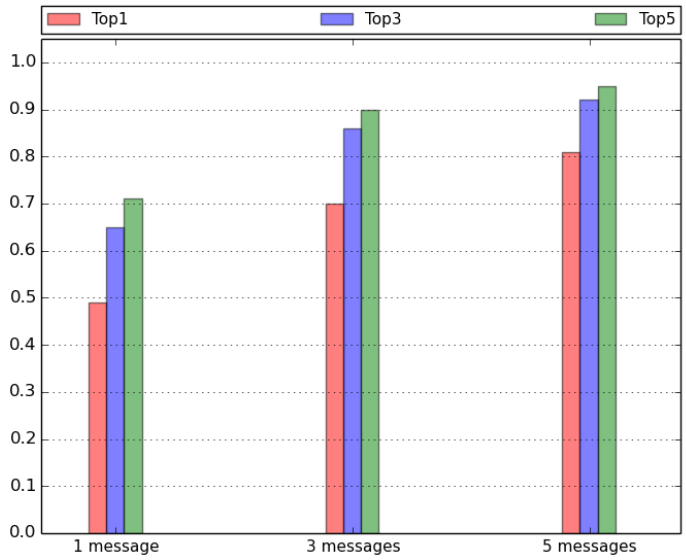


Figure 4.3: Bar chart of the results for character n-grams using single messages and combinations of 3 or 5 messages as input. The bars represent how often the correct user is in the top 1, 3 or 5 of most possible users.

4.2.2 Dutch

This section discusses the results for the experiments conducted with the Dutch data set. Because of the limited amount of available data, the user post criterium had to be adjusted to include enough users. Consequently, the results of the English and Dutch data sets are not easily comparable. Differences are not necessarily based on language, but can also be explained by the smaller amount of data. That is, for the English data set, users that posted

between 75 and 200 messages are included, while for the Dutch data set users this post criterium is lowered to between 25 and 100, due to the limited amount of data. For the Dutch data set character n-grams also deliver the best performance. The classifier is able to select the correct user from 25 different users for 40% of the test instances. In 71% of the cases, the correct user is in the top 5 of most possible users. The smaller amount of data seems to effect the combinations of messages the most. When combining 5 messages, these scores increase to 65% for the top 1 and 89% for the top 5.

Single messages

This first experiments entail investigating the performance of the different feature set using single messages. This section shows that character n-grams are also able to outperform the baseline when Dutch data is used. Using character n-grams, the classifier is able to select the correct author for 40% of the messages from a list of 25 authors. For 69% of the messages the correct author is in the top 5 of most possible authors.

Baseline

Similar to the experiments carried out using the English data, for Dutch the baseline set using character features as well.

	Dutch	
	Mean	(SD)
Top 1	0.19	(0.16)
Top 3	0.38	(0.16)
Top 5	0.49	(0.16)

Table 4.10: Percentages and their variance scores of cases where the correct user is in the top 1, 3 or 5 of most possible users, using character features.

Table 4.10 shows that character features return an accuracy of .19. For 38% of the messages the correct user is in the top 3, and for 49% the correct user is in the top 5. Although this means that in less than 50% of the cases the correct author is seen as one of the 5 most possible author, it does support the hypothesis that it is more likely that the correct author is among a top-n of most possible users. However, where for the English data set a position in the top 5 means the author is in the top 7.5% of most possible users, for the Dutch data set it means a position in the top 20%.

Experiments with different feature sets

Figure 4.4 shows that character n-grams are also able to outperform the baseline when applied to Dutch data and return the highest scores.

While the baseline was able to return an accuracy score of 0.19 (SD=0.16), character n-grams increase this score to 0.40 (SD=0.18). This difference is highly significant, t(113)=8.75, $p < .001$. For the top 3 of most possible authors, the score increases from 0.38 (SD=*0.16*) for the baseline to 0.60 for character n-grams (SD=0.16), which is also a highly significant difference, t(113)=9.34, $p < .001$. For the top 5 of most possible authors, character n-grams

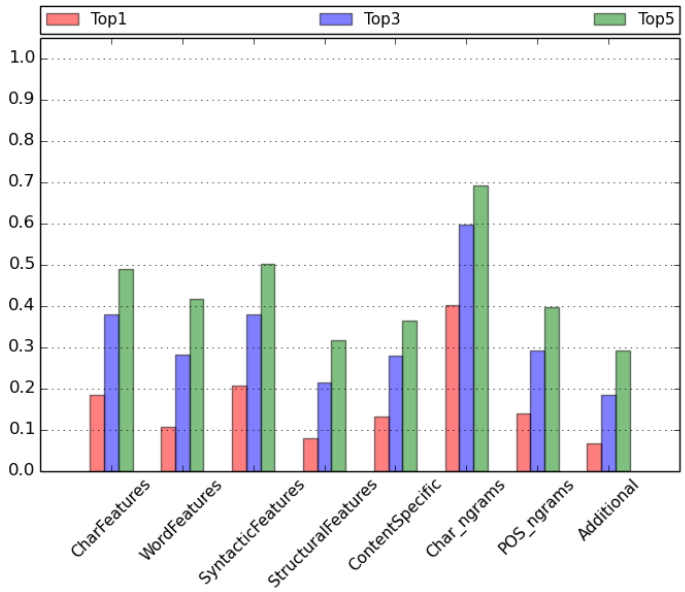


Figure 4.4: Ranking results for the Dutch data set using single messages as input

also score significantly higher (M=0.69, SD=0.14) than the baseline (M=0.49, SD=0.16), t(113)=7.60, $p < .001$.

Combining character n-grams with other feature sets results for the Dutch data set that is mostly similar to the results for the English data set. The differences between using only character n-grams as features and combinations of features are small. However, there are also some combinations that behave differently for Dutch than for English. For example, for English the combination of character n-grams and POS n-grams returned significantly lower scores, while for Dutch no significant difference is found. This is most likely due to fact that the English data set contains much more character n-grams and POS n-grams than the Dutch data set. While the English data set contains 18,909 different character n-grams and 10,787 POS-n-grams, the Dutch data set contains only 4,117 different character n-grams and 1,441 different POS n-grams. Adding 10,787 features to 18,909 features could introduce more confusion to the classifier than adding 1,441 features to 4,117.

Table 4.11 shows the small differences between the scores of the different combinations of feature sets. There are two scores that differ significantly from using only character n-grams. For the top 3, the combination of character n-grams and syntactic features (M=0.62, SD=0.19) differs significantly from the top 3 for only character n-grams (M=0.61, SD=0.19), t(113)=2.30, $p < .05$. For the top 4 only the only combination that is able to significantly outperform only character n-grams (M=0.71, SD=0.17) is with additional features (M=0.72, SD=0.15), t(113)=2.34, $p < .05$.

Combinations of messages

Paragraph 4.2.1 showed that for English the performance increased when combinations of messages rather than single messages were used as input. Therefore, the same experiments are conducted using the Dutch data. That is, for each feature set it is tested if the performance increases when in stead of single messages, average feature values of combinations of 3 or 5 messages are used as input.

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
Char n-grams	0.41	(0.17)	0.61	(0.18)	0.71	(0.17)
Ngrams+Char	0.41	(0.18)	0.61	(0.18)	0.71	(0.16)
Ngrams+Word	0.41	(0.18)	0.61	(0.18)	0.71	(0.16)
Ngrams+Synt	0.41	(0.18)	0.62*	(0.19)	0.71	(0.18)
Ngrams+Stru	0.41	(0.18)	0.61	(0.18)	0.71	(0.16)
Ngrams+Cont	0.41	(0.17)	0.61	(0.18)	0.71	(0.17)
Ngrams+POS_	0.42	(0.18)	0.62	(0.18)	0.71	(0.17)
Ngrams+Addi	0.42	(0.17)	0.62	(0.17)	0.72*	(0.15)

Table 4.11: Results for combinations of character n-grams with other feature sets using single message as input, $^*p < .05$

Because character n-grams return the best results for both single messages as combinations of messages, only the results for that feature set are included below.[1]

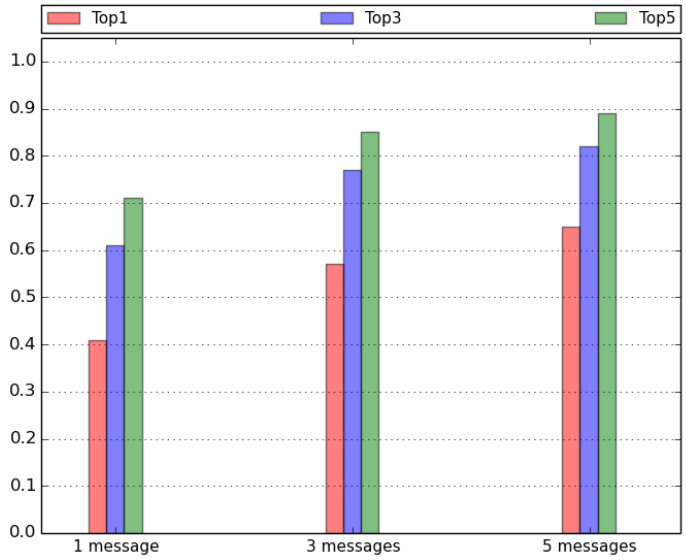


Figure 4.5: Bar chart of the results for character n-grams using single messages and combinations of 3 or 5 messages as input. The bars represent how often the correct user is in the top 1, 3 or 5 of most possible users.

Figure 4.5 gives a visual representation of the improvement when combinations of messages are used. It shows that not only the score for the top 1 improves, but also for the top 3 and 5. Table 4.12 shows a more detailed overview of the results. It shows that combining three messages results in an accuracy score of 0.57 (SD=0.04), as opposed to an accuracy score of 0.41 (SD=0.01) for single messages. Combining five messages results in an accuracy score of 0.65 (SD=0.05).

However, not only the scores increase; the standard deviations increase as well. Using

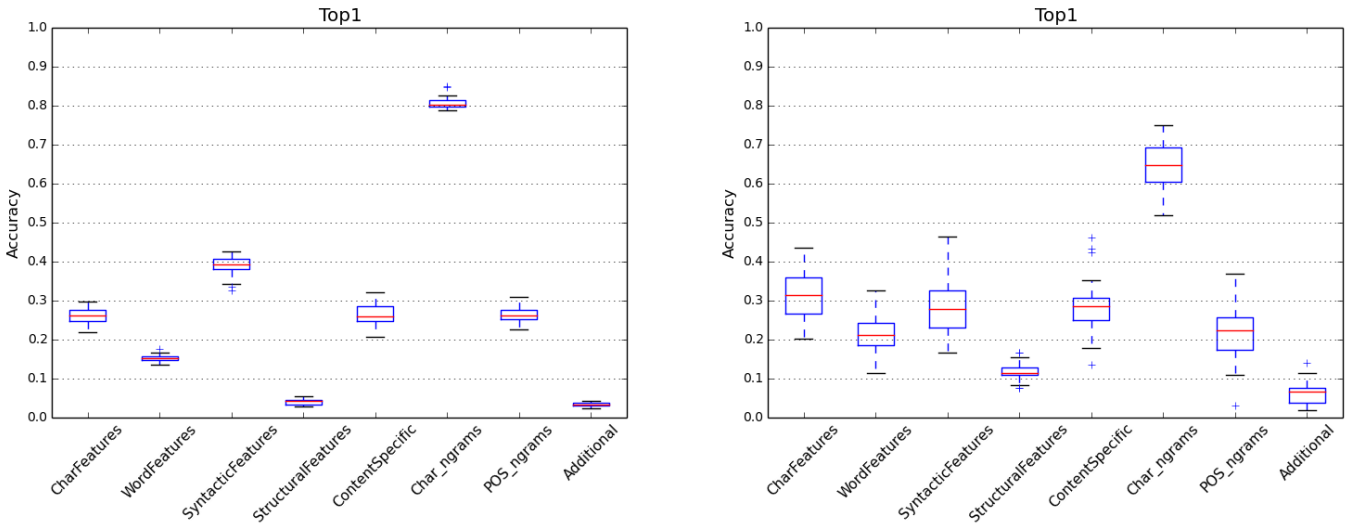
[1]See appendix E for an overview for the results of all feature sets.

	Top1		Top3		Top5	
	Mean	(SD)	Mean	(SD)	Mean	(SD)
1 message	0.41	(0.01)	0.61	(0.02)	0.71	(0.02)
3 messages	0.57	(0.04)	0.77	(0.03)	0.85	(0.03)
5 messages	0.65	(0.05)	0.82	(0.05)	0.89	(0.05)

Table 4.12: Using average feature values of combinations of messages improves the performance of character n-grams

single messages results in a stable system, with a standard deviation not higher than 0.02. For combinations of 3 messages the standard deviation increases to 0.04 and for combination of 5 messages it further increases to 0.05. This indicates that the different combinations of messages do influence the performance of the system.

Although the results from the English and Dutch data sets are not one-to-one comparable, it is interesting to see how the results differ. Figure 4.6 shows the results of the different feature sets when combinations of 5 messages are used for the English data set (figure 4.6a) and for the Dutch data set (figure 4.6b).



(a) English

(b) Dutch

Figure 4.6: Compared to English, the Dutch data set returns lowers scores and larger variation between scores when combinations of 5 messages are used as input

It shows that the relative performance of the different feature sets is comparable for the English and Dutch data set. Character n-grams deliver the best results, and structural and additional features the worst. What is more interesting, however, is that the variation between runs is larger for the Dutch data set than for the English data set. This is most probably due to the fact that the number of messages per user was lower for the Dutch data set than for the English data set. Because there was only a limited amount of data available for Dutch, the post count criterium of 75 for English was lowered to 25 for Dutch.

4.2.3 Preliminary conclusion problem 2

The section above discusses the results for the second problem, that focuses on attribution the correct user to a forum post. The English data set contained 67 users and the Dutch data set 25 users. In order to do that, we investigated which feature set delivers the best performance using SVM as classification algorithm. The hypothesis was that, due to the large number of users and the short length of the forum messages, it was more likely that the correct user is in a top-n of most possible users than that we were able to select the correct user for each post. We did not only look at how the system performed using single messages as input, but also using average feature values of combinations of messages as input. To see if the performance of the different feature sets is similar for Dutch, the experiments are repeated using a Dutch data set.

For English, character n-grams deliver the best performance. Using single message as input results in an accuracy score of 0.49. For 72% of the cases, the correct user is in the top 5 of most possible users. Using combinations of 5 messages improves these results. The accuracy score increases to 0.81 and in 95% of the cases, the correct user is in the top 5 of most possible users. This suggests that this approach is very useful to attribute the correct user to a form post.

Character n-grams also deliver the best performance for Dutch. For single messages, the accuracy score was 0.40 and the correct user was in the top 5 of most possible users in 71% of the cases. Combining messages again increased these scores, but not as much as for the English data set. Using combinations of 5 messages increased the accuracy score to 0.65 and for 89% of the messages, the correct user was in the top 5. However, using combinations did not only increase the scores, but also the variance. This is most likely due to the limited amount of data available.

Chapter 5

Discussion

This research investigated to what extent we are capable of identifying individuals by their typical writing style on anonymous web forums. In order to do that we conducted experiments for two different problems: The first problem considered whether two sets are written by the same or by different users and the second problem considered attributing a forum post to the correct author. For both problems, the outcomes are very promising and suggest that we are indeed able to capture enough of a user's writing style to distinguish between several users.

Before focusing on the findings for the two different problems more thoroughly, there are some remarks that are applicable to both problems. The first remark is that, for both problems, character n-grams perform really well and outperform all other feature sets. For the first problem, using character *n-grams* results an F-score of 0.89, indicating that the system is performs very well in classifying whether two sets of texts are written by the same or by different users. For the second problem, character n-grams achieve an accuracy score of 0.49 for single messages, but this increases to 0.81 when combinations of messages are used.

The good performance of character n-grams are in line with other works on authorship attribution (e.g. Forsyth and Holmes, 1996; Kešelj et al., 2003; Stamatatos, 2008). Stamatatos (2009), for example, points out the robustness of n-grams and Cavnar et al. (1994) adds to this by pointing out that an n-gram-based approach is able to deal with textual errors. As forum messages contain many textual errors, this is a particularly helpful attribute. Another reason for why character n-grams outperform all other feature sets is that they implicitly represent a large number of other features. For example, character n-grams include information about topic, punctuation, regular spelling errors by users, but also about the use of function words that prove to perform well for authorship attribution tasks (e.g. Mosteller and Wallace, 1963; Burrows, 1989; Argamon and Levitan, 2005).

Using n-grams over other feature sets has several advantages. First of all, n-grams are language independent. For example, n-grams outperform other feature sets for authorship attribution in English, Greek and Chinese (Kešelj et al., 2003; Peng et al., 2003). A second advantage is that for n-grams, no part of speech tagging is necessary. Especially for short and informal texts, such as forum messages, automated part of speech tagging proves difficult. Another advantage is that n-grams also implicitly includes the stemming of words. As Cavnar et al. (1994) point out, the n-grams for related word forms (e.g. 'advance', 'advanced', 'advancing') are very similar.

What we did not investigate, however, is which specific features are mainly responsible for this outcome. It would be interesting to see whether or not using a subset of features

could increase the performance. Cavnar et al. (1994) show that the frequency of n-grams correlate to text characteristics. They state that "the top 300 [most occuring] or so N-grams are almost always highly correlated to the language", and that "starting around rank 300 or so, an N-gram frequency profile begins to show N-grams that are more specific to the subject of the document" (Cavnar et al., 1994, pp. 163-164). It would be interesting to investigate if n-grams from a specific rank correlate to authorship. Moreover, future work should consider selecting only the features that perform best in distinguishing between authors, and thereby neglecting features that distinguish, for example, between languages or subject.

Another remark is that the results show that combining multiple messages into a single training instance improves the system's performance. While for the first problem a rather drastic combination is made, combining half of a user's messages, the second problem showed that combining only 3 or 5 messages already leads to an increase in performance. One potential reason why this works so well, is that a combination of messages is better able to capture the writing style of an author, ironing out anomalous messages, rather than capturing the writing style of a single message. For example, small messages like thank you notes are less likely to cause confusion when they are combined with other messages with more content. What is not clear however, is how many messages are needed in order to obtain the optimal performance.

5.1 Problem 1: One user with multiple aliases

The first problem dealt with users that operate under two user names. The resulting system aims to answer the following question: Are these text written by the same or by different users? For this problem two different approaches were employed. The first approach tried to reconstruct the users as good as possible by using the clustering algorithm k-means. As discussed more thoroughly in 5.1.1, this approach did not yield very promising results. In our second approach we employed a classifier to decide whether or not two sets of messages are written by the same user.

5.1.1 Clustering

The aim of this approach was to define whether or not the the clustering algorithm k-means can distinguish between two users using stylometric features. The hypothesis was that if two user names belong to the same user on the one hand, a clustering algorithm would not be able to create two clearly defined clusters. On the other hand the hypothesis was that if two user names belong to different users, a clustering algorithm would be able to create two clearly defined clusters. The results, however, do not indicate that any of the feature sets is able to create more clearly defined clusters of messages based on authorship when messages from two different users are used as input than when messages from a single user are used as input.

Figure 5.1 shows a scatter plot of the results of data from two users, using word features, for example: number of short and long words, average word length and number of words occurring once or twice. The image on the left shows how the clustering algorithm clusters the messages, or data points, from two different users. One user represented by the stars, and another user by the circles. It shows a strong of overlap of the data points of both users. The image on the right shows that the clustering algorithm did define two clusters, but that this is can not be accounted for by authorship

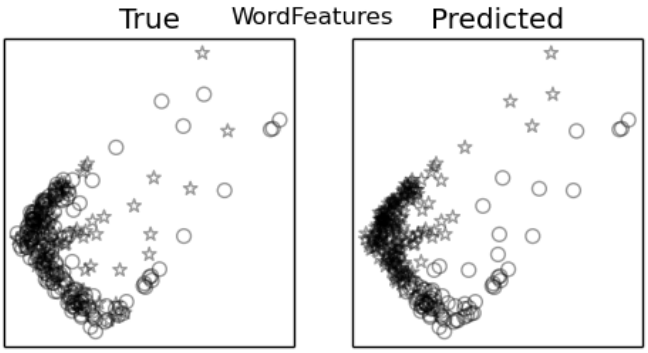


Figure 5.1: Scatter plots for the results of k-means using word features. The image on the left shows the true labels (a star represents one user, and a circle the other) and the image on the right shows the predicted labels.

Another problem that appears with the clustering task, is that the results differs for each set of users. For example, while using word features for one set of users results in a V-measure of 0.184, for another set of users it yields a V-measure of only 0.005. This indicates that the scores are not only low, they are also inconsistent, as they fluctuate depending on the specific users that are used as input. One possible reason for this is that the clustering could actually be based on, for example, topic. This would mean that the results fluctuate depending on whether or not the users write about the same topic. That is, if the clustering algorithm is applied to two users that write about different topics (e.g., one user generally writes only about drugs, and the other about weapons), the results are better than when the algorithm is applied to two users that write about the same topic.

In contrast with these results, Novak et al. (2004) obtain promising results using a clustering algorithm for a similarity detection task. However, they also show the negative influence of topic. Their best results are retrieved with a data set containing messages about the same topic, and when they introduce a variety of topics their results decrease. Because our data set is not filtered for topic, this is in line with the low results we obtained. Not only Novak et al. (2004) notice the influence of topic, Stamatatos (2009, pp. 552) also points out that "ideally, all the texts of the training corpus should be on exactly the same topic for all the candidate authors". Therefore, additional research should be done using data that considers only one topic. However, as Novak et al. (2004) also show, messages that are slightly off-topic can prove problematic. They propose combining text classification with their clustering approach to deal with off-topic messages.

Something worth investigating is how using combinations of messages instead of single messages influences the results of the clustering algorithm. Using a more generalized model that looks at differences between the average feature values of half of the user's messages to decide whether two sets of texts are written by the same or by different users delivers promising results, as shown in section 4.1.2. This approach, however, is not an option to use together with an clustering algorithm, as it would result in only two instances per user. However, the results for the second problem, that focuses on attributing the right author to a forum message, show that the results already improve significantly using average feature values of three or five messages instead of single messages. Future investigation should point out if this method is capable of improving the clustering results.

5.1.2 Classifying

Because the clustering approach discussed above did not yield very promising results, another approach is considered. The aim is similar: the system should be able to answer the question whether two sets of messages are written by the same or by different authors. For this approach we split the messages of the authors into two sets and calculated the average feature values for each set of messages. This resulted in two sets of feature vectors (A_1, B_1, $C_1 \ldots N_1$) and (A_2, B_2, C_2, $\ldots N_2$). Each author's feature vector from the first set was then compared to the second feature vector of the same user (e.g. A_1 with A_2) to create a feature vector representing messages from the same user, and to the second feature vector of different users (e.g. A_1 to $B_2 \ldots N_2$) to create a feature vector representing messages from different users. These vectors were used to train a binary classifier that returns whether or not two sets of messages are written by the same user.

Section 4.1.2 showed that this approach returns very promising results for distinguishing between two sets of messages written by the same user and two sets of messages written by different users. Using character n-grams, the classifier returned an F-score of .89. A closer inspection of the scores showed that the high F-score was mainly accounted for by the precision score, as this score was very high for both classes (0.94 and 0.98). The recall score, however, was only very high for the class for messages from different authors (1.0). The recall score for the class for messages from the same author was relatively low (0.68). This means that of sets of messages written by the same user, only 68% of the instances retrieved. For an application in forensics, a low recall score is less problematic than a low precision score would be, as it is more important that the classifier does not make any mistakes in deciding that two sets of texts are written by the same author. If it would make mistakes, it could lead to false accusations.

Although not all the sets of texts written by the same user are retrieved, these results are very promising. They suggest that a binary classification task using character n-grams as features is very useful for distinguishing between two sets of texts written by the same author and sets of texts written by different authors. To find an explanation for these promising results, the feature vector for character n-grams is empirically analyzed.

Figure 5.2 gives a general idea of the distribution of feature values. This distribution shows that for the class label *same* the values of these features are mostly lower than the values of the class label *different*. A similar pattern occurs for many other instances and features not shown by this excerpt, but not all. The fact that this pattern occurs for many other instances indicates that there is enough ground for the high precision score. The fact that this pattern does not occur for all instances and features, however, might be a cause for the relatively low recall score.

Another explanation for this good performance could be that using average feature values of combinations of messages smoothens the data. Rather than having multiple feature vectors of one message to represent a user, a feature vector of half of a user's messages is used as representation. In contrast to the clustering effect, short or anomalous messages, such as thank you notes, do no longer interfere because they are combined with other messages.

Overall, the results indicate that this method provides a topic-independent solution that is able to distinguish sets of texts written by the same user from sets of texts written by different users. Moreover, this solution does not depend on differences between specific users, but depends on differences between users in general. That is, the system uses the variances between two sets of messages from one set of users to decide whether two sets of messages

variance / class	*n*-gram 1	n-gram 2	n-gram 3	n-gram 4	n-gram 5	n-gram 6	n-gram 7
same (A_1+A_2)	**4.00E-05**	**1.00E-05**	**2.00E-05**	**0**	**7.00E-05**	**0.00056**	**7.00E-05**
different (A_1+B_1)	1.00E-05	0	9.00E-05	0.00105	0.00055	0.00044	0.00043
different (A_1+C_1)	0.00025	1.00E-05	9.00E-05	0.00085	0.00149	0.00062	0.00025
different (A_1+D_1)	0.00018	2.00E-05	0.00056	0.00117	0.00022	0.0006	0.00043
different (A_1+E_1)	0.00029	0	0.00296	0.00117	0.00099	0.00187	5.00E-05
different (A_1+F_1)	0.00044	0	0	0.00087	0.00103	0.00152	0.00023
same (B_1+B_2)	**0.00E+00**	**0.00E+00**	**0.08684**	**0**	**0.11818**	**0**	**0**
different (B_1+G_1)	0	1.69E-01	0.32564	8.78E-02	0.43072	0	0
different (B_1+H_1)	0	0.00E+00	0.32564	0.00E+00	0.22852	0.59063	0
different (B_1+I_1)	0	0.00E+00	0.31294	0	0.96809	0.87128	0.06147
different (B_1+J_1)	7.67E-01	7.78E-01	0.76949	0	0	0	0.13437
different (B_1+K_1)	0.00E+00	0.41176	8.43E-01	0	0.14338	0	0.13437

Figure 5.2: Excerpt of the feature vector for character *n*-grams of the variance between messages of the same user (between A_1 and A_2) and different users (between A_1 and B_2, C_2 ... N_2).

are written by the same or by different users. One thing that is not clear and should be investigated, however, is how the size of the data set influences the results. How many messages, or words, are needed per set of messages in order to obtain the best performance?

One of the shortcomings of this research is that we only investigated if combining character *n*-grams with other feature sets would lead to an improvement of the recall score, which it did not. What we did not investigate, is if the recall score would improve if less features were used, that is, if we used a feature selection algorithm to select only the best performing features. This would reduce the dimensionality of the feature vector, and may lead to less confusion of the system.

A second shortcoming is that, although this approach performs really well on our data, it is not clear how well the system performs in real situations. If a user feels the need to use multiple aliases on a forum, it is expected that he or she[1] uses them for different purposes. For example, he might use one alias to promote his products, while he uses another to post reviews about how trustworthy his other account is. This would result in two very different types of messages, and future work should focus on how this influences the performance of the system. The next step would be to find out how well the system performs when messages from multiple sources are used. Would the system be able to detect two accounts of the same user on two different forums, or on Facebook and a forum? Additionally, following Johansson et al. (2013), it would be interesting for both issues how using time profiles as feature influence the performance. This could add an additional dimension as opposed to only using linguistic data.

[1]for the sake of clarity, from now on only *he* is used

5.2 Problem 2: User ranking for unseen posts

The second problem focused on attributing a specific author to a previously unseen messages. Therefore, a classifier was trained with one part of the messages, and the other messages were used to test the performance. The hypothesis was that it is more likely that the system would be able to select a top-n of potential authors, instead of selecting a single author. There were two main outcomes. The first outcome is that character n-grams again outperform all other feature sets. For the English data set, n-grams resulted in an accuracy score of 0.50 for the top 1, 0.66 for the top 3 and 0.72 for the top 5.[2] Using average feature values of combinations of messages rather than feature values of single messages improves the results. For English, the best results are obtained with combinations of 5 messages, that results in an accuracy score of 0.81 for the top 1, 0.92 for the top 3 and 0.95 for the top 5. For the Dutch data set, n-grams returned lower scores, but it was still the best performing feature set. Using single messages n-grams returned an accuracy score of 0.41 for the top 1, 0.61 for the top 3 and 0.71 for the top 5. Using combinations of 5 messages increased these scores to 0.65 for the top 1, 0.82 for the top 3 and 0.89 for the top 5.

These results support the hypothesis, as all scores for the top 3 and top 5 are higher than for the top 1. This indicates that when the system is not able to assign the correct author to a messages, the author is among the top 3 or 5 of most possible users. When combinations of 5 messages are used, the correct author is in the top 5 for 95% of the instances. This implies that the system is very capable of selecting the top-n of most possible users. For Dutch, however, the scores are lower. It is difficult to compare the results for the English and Dutch data, because for Dutch only a limited amount of data was available. This could make it more difficult to capture enough of the writing style of an author. Although there was less data per author available, there were also less authors (67 for English and 25 for Dutch), which could actually make the problem easier. In order to be able to compare the results for the English and Dutch data, future research should use similar data-sets.

Using combinations of messages instead of single messages as input is related to the solution Stamatatos (2008) proposes to handle class-imbalances, aiming to solve the skewness of the distribution of number of texts per author. As a solution, he re-samples the texts so that each has the same number of texts. This means that the minority classes get short text samples and the majority classes get larger text-samples. For this research it would mean that not only the number of samples is the same per user, but also that, for example, thank you notes, would be combined with other messages.

A downside of this approach is that it increases the number of messages needed in order to obtain stable results. Although it is unclear what the exact number of messages is, the differences between the results of the English and Dutch data imply that a lower number of messages decreases the results and increases the variance between the results. This suggests that this method becomes more unreliable when less data is used. It would be interesting to see how the system performs using an English data set with the same number of uses and posts as we used for Dutch. The influence of the amount of data relates to the issue Luyckx and Daelemans (2008) point out. Their issue is that a lot of studies considering authorship attribution use "unrealistic sizes of training data", with over 10,000 words per author. In more realistic situations, and particularly in forensics, there is often only limited training data. To

[2] top 1 means that the correct author is the most possible author, and top 3 and 5 that the correct author is in the top n of most possible authors.

illustrate, they show the positive influence of increasing the data size on the performance.

For future research it is important to focus more specifically on how corpus size influences the performance. A question would be how the findings relate to the findings of Luyckx and Daelemans (2008) and how much training data is needed to return satisfactory results. To date, there is no clear threshold for text-length (Stamatatos, 2009). Hirst and Feiguina (2007), for example, report promising results with texts of less than 1,000 words. However, as Luyckx and Daelemans (2008) point out, the total size of their corpus is still unrealistic, due to the large number of texts. Using combinations of messages, as investigated in this thesis, or using a re-sampling approach that makes even-sized chunks (Stamatatos, 2008), may decrease the influence of text length on the one hand, but on the other hand it may increase the influence of the number of texts used.

Chapter 6

Conclusion

In this thesis we investigated the possibilities for authorship detection of forum posts on anonymous web forums, based on their writing style. One of the main aims was to find which stylometric features or combination of features work well to attribute the correct author to a forum post. As a starting point we used the framework proposed by Zheng et al. (2006), that includes feature sets based on characters, words, function words, structure and content. Character n-grams and POS-trigrams were added to this framework. To test the performance of the different features, the investigation was split up into two different problems: the first problem aimed to detect users with multiple aliases and the second to attribute an author to an unseen post.

For the first problem we showed that the clustering approach k-means with single messages as input did not deliver very satisfying results. Clusters were not formed based on authorship, but most probably on topic or function of a text. Therefore, another approach was implemented. Rather than using single messages as input, feature vectors were created using a set containing half of a user's messages. The final feature vectors, used to train an SVM-classifier, consisted of the variance between sets from the same user, with the class label *same*, and the variance between sets from different users, with the class label *different*. With this approach, and character n-grams as features, we were able to identify sets of texts from the same user with an F-score of .89. By combining half of a user's messages into one feature vector, the influence of topic is reduced.

For the second problem we investigated how often the correct user is in the top-n of most possible authors. Using character n-grams as features and single messages as features, the SVM-classifier obtains an accuracy score of .50 for the English data set. For 72% of the messages, the correct user is in the top 5 of most possible users. We also showed that using average feature values of combinations of messages has a positive effect on the scores, increasing the accuracy to .81. This indicates that combining a smaller number of messages also reduces the influence of topic. A downside of this approach, however, is that it increases the number of messages needed to deliver a stable performance. As the experiments conducted using the smaller Dutch data set show, using combinations of five messages makes the system less stable.

To conclude, the above shows that writing style can indeed be very helpful for identifying individuals on anonymous web forums by their typical writing style. Especially character n-grams perform very well, outperforming all other feature sets. Using combinations of messages instead of single messages as input reduces the influence of the topic of the post, enabling us to create a more topic-independent system.

Bibliography

Abbasi, A. and Chen, H. (2006). Visualizing authorship for identification. In *Intelligence and Security Informatics*, pages 60–71. Springer.

Abbasi, A. and Chen, H. (2008). Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, 26(2):7.

Afroz, S., Brennan, M., and Greenstadt, R. (2012). Detecting hoaxes, frauds, and deception in writing style online. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 461–475. IEEE.

Argamon, S. and Levitan, S. (2005). Measuring the usefulness of function words for authorship attribution. In *ACH/ALLC*.

Argamon, S., Šarić, M., and Stein, S. S. (2003). Style mining of electronic messages for multiple authorship discrimination: first results. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 475–480. ACM.

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

Baayen, H., van Halteren, H., Neijt, A., and Tweedie, F. (2002). An experiment in authorship attribution. In *6th JADT*, pages 29–37. Citeseer.

Baayen, H., Van Halteren, H., and Tweedie, F. (1996). Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing*, 11(3):121–132.

Bennett, K. P. and Campbell, C. (2000). Support vector machines: hype or hallelujah? *ACM SIGKDD Explorations Newsletter*, 2(2):1–13.

Bu, Z., Xia, Z., and Wang, J. (2013). A sock puppet detection algorithm on virtual spaces. *Knowledge-Based Systems*, 37:366–377.

Burrows, J. F. (1989). an ocean where each kind...: Statistical analysis and some major determinants of literary style. *Computers and the Humanities*, 23(4-5):309–321.

Cavnar, W. B., Trenkle, J. M., et al. (1994). N-gram-based text categorization. *Ann Arbor MI*, 48113(2):161–175.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

De Smedt, T. and Daelemans, W. (2012). Pattern for python. *The Journal of Machine Learning Research*, 13(1):2063–2067.

De Vel, O. (2000). Mining e-mail authorship. In *Proc. Workshop on Text Mining, ACM International Conference on Knowledge Discovery and Data Mining (KDD2000)*.

De Vel, O., Anderson, A., Corney, M., and Mohay, G. (2001). Mining e-mail content for author identification forensics. *ACM Sigmod Record*, 30(4):55–64.

Diederich, J., Kindermann, J., Leopold, E., and Paass, G. (2000). Authorship attribution with support vector machines. In *APPLIED INTELLIGENCE*. Citeseer.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Forsyth, R. S. and Holmes, D. I. (1996). Feature-finding for text classification. *Literary and Linguistic Computing*, 11(4):163–174.

García, A. M. and Martin, J. C. (2006). Function words in authorship attribution studies. *Literary and Linguistic Computing*.

Haeseryn, W., Romijn, K., Geerts, G., De Rooij, J., and van den Toorn, M. (1997). Algemene nederlandse spraakkunst. groningen/deurne: Martinus nijhoff uitgevers. retrieved from http://ans.ruhosting.nl/e-ans/index.html, 23 June 2014.

Hirst, G. and Feiguina, O. (2007). Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*, 22(4):405–417.

Holmes, D. I. (1994). Authorship attribution. *Computers and the Humanities*, 28(2):87–106.

Holmes, D. I. (1998). The evolution of stylometry in humanities scholarship. *Literary and linguistic computing*, 13(3):111–117.

Johansson, F., Kaati, L., and Shrestha, A. (2013). Detecting multiple aliases in social media. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1004–1011. ACM.

Kešelj, V., Peng, F., Cercone, N., and Thomas, C. (2003). N-gram-based author profiles for authorship attribution. In *Proceedings of the conference pacific association for computational linguistics, PACLING*, volume 3, pages 255–264.

Koppel, M., Schler, J., and Argamon, S. (2009). Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology*, 60(1):9–26.

Layton, R., Watters, P., and Dazeley, R. (2010). Authorship attribution for twitter in 140 characters or less. In *Cybercrime and Trustworthy Computing Workshop (CTC), 2010 Second*, pages 1–8. IEEE.

Ledger, G. and Merriam, T. (1994). Shakespeare, fletcher, and the two noble kinsmen. *Literary and Linguistic Computing*, 9(3):235–248.

Luyckx, K. and Daelemans, W. (2008). Authorship attribution and verification with many authors and limited data. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 513–520. Association for Computational Linguistics.

Mendenhall, T. C. (1887). The characteristic curves of composition. *Science*, (214S):237–246.

Mosteller, F. and Wallace, D. L. (1963). Inference in an authorship problem: A comparative study of discrimination methods applied to the authorship of the disputed federalist papers. *Journal of the American Statistical Association*, 58(302):275–309.

Novak, J., Raghavan, P., and Tomkins, A. (2004). Anti-aliasing on the web. In *Proceedings of the 13th international conference on World Wide Web*, pages 30–39. ACM.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peng, F., Schuurmans, D., Wang, S., and Keselj, V. (2003). Language independent authorship attribution using character level language models. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 267–274. Association for Computational Linguistics.

Rosenberg, A. and Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420. Citeseer.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Santorini, B. (1990). Part-of-speech tagging guidelines for the penn treebank project (3rd revision).

Solorio, T., Pillay, S., Raghavan, S., and Montes-y Gómez, M. (2011). Modality specific meta features for authorship attribution in web forum posts. In *IJCNLP*, pages 156–164.

Stamatatos, E. (2008). Author identification: Using text sampling to handle the class imbalance problem. *Information Processing & Management*, 44(2):790–799.

Stamatatos, E. (2009). A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556.

Tallentire, D. R. (1973). Towards an archive of lexical norms: A proposal. *The computer and literary studies*, pages 39–60.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). 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-Volume 1*, pages 173–180. Association for Computational Linguistics.

Van den Bosch, A., Busser, B., Canisius, S., and Daelemans, W. (2007). An efficient memory-based morphosyntactic tagger and parser for dutch. In *Computational Linguistics in the Netherlands: Selected Papers from the Seventeenth CLIN Meeting*, pages 99–114.

Van Eynde, F. (2001). Part of speech tagging en lemmatisering. *Openbaar CGN-document (Publicly available CGN document), KU Leuven*.

Zheng, R., Li, J., Chen, H., and Huang, Z. (2006). 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.

Appendix A

CGN to Penn tag mapping

CGN	Penn
ADJ\(.*,basis.*\)	JJ
ADJ\(.*,comp.*\)	JJR
ADJ\(.*,sup.*\)	JJS
ADJ\(dial\)	JJ
ADJ\(vrij,(basis|dim).*\)	RB
ADJ\(vrij,comp,zonder\)	RBR
ADJ\(vrij,sup,zonder\)	RBS
BW\(\)	RP
LET\(\)	SYM
LID\(.*\)	DT
N\(eigen,ev.*\)	NNP
N\(eigen,mv.*\)	NNPS
N\(soort,ev.*\)	NN
N\(soort,mv.*\)	NNS
SPEC\(afk\)	ABB
SPEC\(deeleigen\)	NNP
SPEC\(symb\)	SYM
SPEC\(vreemd\)	FW
TSW\(\)	UH
TW\(hoofd.*\)	CD
TW\(rang.*\)	JJ
VG\(neven\)	CC
VG\(onder\)	IN
VNW\(.*,adv-pron,(gen—stan),red,3,getal\)	EX
VNW\(.*,adv-pron,obl.*\)	RB
VNW\((aanw|betr|pr|pers|onbep|vb|vrag|refl|recip),(pron|det).*\)	PRP
VNW\(bez,.*\)	PRP$
VNW\(onbep,grad.*\)	CD
VNW\(vb,det,.*\)	WDT
VNW\(vb,pron,gen.*\)	WP$
VZ\(.*\)	IN
WW\((inf|pv,conj).*\)	VB

WW\(od,.*\)	VBG
WW\(pv,tgw,met-t\)	VBZ
WW\((pv,tgw,(ev|mv)|dial)\)	VBP
WW\(pv,verl.*\)	VDB
WW\(vd,.*\)	VBN

Table A.1: Regular expressions used for the mapping from CGN tag labels to Penn tag labels

Appendix B

Function words

B.1 English function words

a	between	in	nor	some	upon
about	both	including	nothing	somebody	us
above	but	inside	of	someone	used
after	by	into	off	something	via
all	can	is	on	such	we
although	cos	it	once	than	what
am	do	its	one	that	whatever
among	down	latter	onto	the	when
an	each	less	opposite	their	where
and	either	like	or	them	whether
another	enough	little	our	these	which
any	every	lots	outside	they	while
anybody	everybody	many	over	this	who
anyone	everyone	me	own	those	whoever
anything	everything	more	past	though	whom
are	few	most	per	through	whose
around	following	much	plenty	till	will
as	for	must	plus	to	with
at	from	my	regarding	toward	within
be	have	near	same	towards	without
because	he	need	several	under	worth
before	her	neither	she	unless	would
behind	him	no	should	unlike	yes
below	i	nobody	since	until	you
beside	if	none	so	up	your

Table B.1: List of English function words

B.2 Dutch function words

hebben	onze	elke	na	voor	zodat
worden	jullie	alle	naar	voorbij	om
zijn	zij	enige	naat	zonder	opdat
zullen	hen	sommige	nabij	dat	indien
de	hun	genoeg	om	of	mits
het	men	voldoende	omtrent	omdat	tenzij
een	die	zat	onder	doordat	hoewel
ik	deze	wat	op	aangezien	alhoewel
me	dit	weinig	over	daardoor	of
mij	dat	weinige	per	door	al
mijn	degene	enkele	qua	met	zonder
je	diegene	aan	richting	dat	behalve
jij	datgene	achter	rondom	voor	uitgezonderd
jou	enig	behoudens	sinds	voordat	naar
jouw	enige	betreffende	te	tot	als
hij	iedereen	bij	tegen	totdat	zoals
hem	iedere	binnen	tegenover	terwijl	gelijk
zich	iemand	boven	tijdens	wanneer	dan
zichzelf	zelf	buiten	tot	zolang	en
zijn	welke	conform	tussen	toen	noch
zij	welk	door	uit	nu	maar
haar	wat	in	van	als	doch
we	wie	inzake	vanaf	nadat	dan
wij	iedere	langs	via	na	want
ons	elk	met	volgens	zodra	dus

Table B.2: List of Dutch function words

Appendix C

Content-specific words

C.1 English words

bmr	shit	backopy	lol	scammer
5onwnspjvuk7cwvk	bitcoins	legit	bitcoin	fucking
scammers	listing&id	scammed	mdma	ive
fec33nz6mhzd54zj	dude	viewtopic	didnt	scamming
paypal	login	haha	r6rcmz6lga4i5vb4	viewuser&id
vpn	mods	google	admins	glock
fucked	encrypt	whats	meth	username
feedbacks	anyways	btw	https	ebay
tormail	deeo	wouldnt	blockchain	blotters
viewuserlistings&id	carding	isnt	silkroad	jpg
asap	mtgox	v5	usb	theres
whitecrane	linux	wtf	ulbricht	xanax
havent	decrypt	1g	buds	demonfifa
ghb	spam	darknet	turd	btcs
blackmarket	100mg	iworks	tormarket	wasnt
lecontog	shitty	newbie	pissed	messaged
asshole	cuz	reship	gnupg	anon
upload	youtube	gox	cashout	javascript
emails	backcopy	moderators	10mg	sr2
msg	counterfeits	inbox	ddos	thx
phishing	idk	bitstamp	benzos	localbitcoins
coderaker	cherryflavor	spamming	blotter	dkn255hz262ypmii
message——hash	youre	ketamine	newbies	privnote
fishy	shrooms	hackbb	truecrypt	2cents
atleast	nqa	hahaha	aswell	organix
pls	wifi	fuckin	sativa	reddit
sheep5u64fi457aw	karma	yep	hologram	kss62ljxtqiqdfuq
bullionaire	darkweb	pid	g0dlike	ruudnl
ssn	signature——version	nope	e-mail	xtc
facebook	codeine	telus	addy	50mg

C.2 Dutch words

bmr	pm	vpn	bitcoins	btc
ff	onion	escrow	vendor	php
5onwnspjvuk7cwvk	paypal	topic	wallet	bitcoin
thanks	listing	mdma	suc6	glock
ruudnl	backopy	cheers	dmt	seller
pgp	mvg	idd	adress	openvpn
carden	opzoek	listings	proxy	cocaine
tormail	sample	oke	maargoed	vendors
belgie	aub	bitstamp	https	scam
trace	listing&id	3v	fe	scammers
scammer	pmk	accounts	belgiumcc	loyaldutchman
gegen	pickel	mittel	mg	2-fma
aanbied	intresse	imei	dutchgoldfish	cf
ideal	tog	tutorial	admin	9mm
javascript	paco	blockchain	gunrunner	lifehacker
p22	verzending	cherryflavor	shipping	iban
walther	vpns	maarja	edit	amfetamine
verzenden	bmk	posts	acount	androids
img	cherry	bijv	stealth	irl
address	mods	f2f	carding	isp
dispute	user	mac-adres	moderator	silk
spoofing	mining	wss	mn	geinteresseerd
iedergeval	drugsmania	hacken	mischien	partitie
hmm	cashen	1x	cosanostra	bitonic
zoraki	mss	xd	acceptgiro	encryptie
escro	phishing	ghash	remote-acces	vraagje
bankaccount	alarmpistolen	methamfetamine	proxpn	coins
riseup	fec33nz6mhzd54zj	ip-adres	prepaid	ekol
kredietkaart	router	profile	none	grtz
frastier	minen	legit	turtlebeach	tomyum

Appendix D

Tables and figures for problem 1

D.1 Problem 1-1: Clustering

D.1.1 Baseline

Users	Homogeneity	Completeness	V-measure
User A_1+A_2	0.003	0.003	0.003
User B_1+B_2	0.02	0.024	0.021
User C_1+C_2	0.029	0.067	0.04

Table D.1: Baseline: clustering scores for single users using character features

Users	Homogeneity	Completeness	V-measure
User A+D	0.003	0.032	0.005
User B+E	0.021	0.029	0.024
User C+F	0.005	0.006	0.005

Table D.2: Baseline: clustering scores for two different users using character features

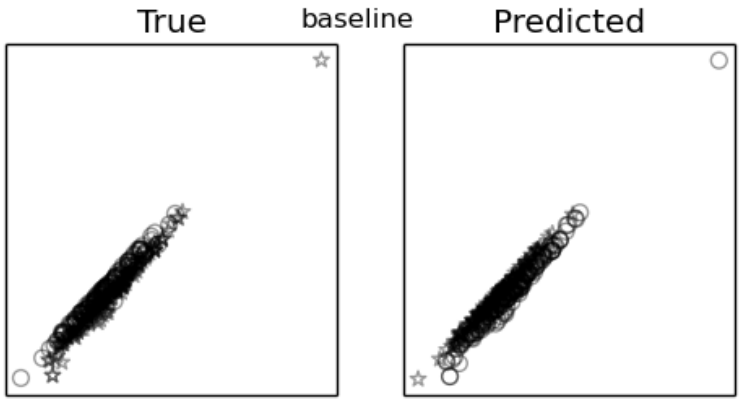


Figure D.1: Scatter plot for the clustering results using character features for users A and D

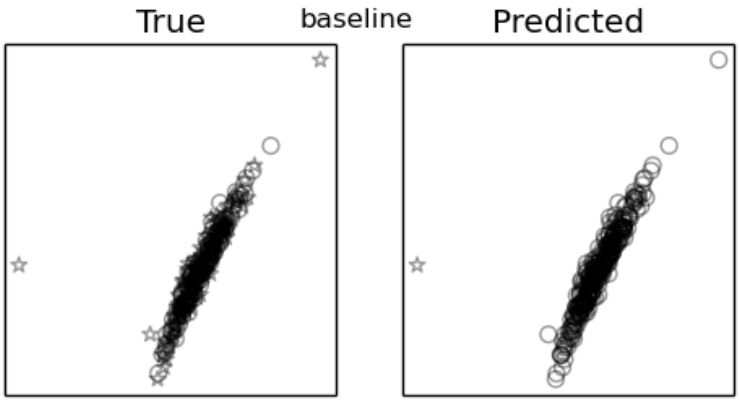


Figure D.2: Scatter plot for the clustering results using character features for users B and E

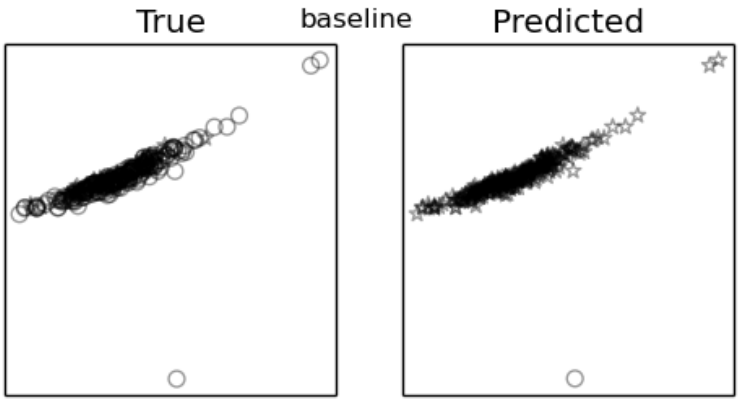


Figure D.3: Scatter plot for the clustering results using character features for C and F

D.1.2 Experiments with different featuresets

One user

Featureset	Homogeneity	Completeness	V-measure
CharFeatures	0.003	0.003	0.003
WordFeatures	0.013	0.014	0.014
SyntacticFeatures	0.016	0.137	0.029
StructuralFeatures	0.037	0.081	0.051
ContentSpecific	0.008	0.12	0.015
Char n-grams	0.011	0.013	0.012
POS n-grams	0.008	0.12	0.015

Table D.3: Clustering results for user A_1+A_2

Featureset	Homogeneity	Completeness	V-measure
CharFeatures	0.02	0.024	0.021
WordFeatures	0.013	0.013	0.013
SyntacticFeatures	0.002	0.002	0.002
StructuralFeatures	0.008	0.12	0.015
ContentSpecific	0.008	0.12	0.015
Char n-grams	0.051	0.071	0.059
POS n-grams	0.008	0.12	0.015

Table D.4: Clustering results for user B_1+B_2

Featureset	Homogeneity	Completeness	V-measure
CharFeatures	0.029	0.067	0.04
WordFeatures	0.0	0.0	0.0
SyntacticFeatures	0.008	0.043	0.013
StructuralFeatures	0.002	0.003	0.002
ContentSpecific	0.003	0.008	0.004
Char n-grams	0.011	0.127	0.02
POS n-grams	0.011	0.127	0.02

Table D.5: Clustering results for user C_1+C_2

Two users

Featureset	Homogeneity	Completeness	V-measure
CharFeatures	0.003	0.032	0.005
WordFeatures	0.164	0.211	0.184
SyntacticFeatures	0.008	0.126	0.015
StructuralFeatures	0.026	0.114	0.043
ContentSpecific	0.034	0.198	0.058
Char n-grams	0.003	0.126	0.006
POS n-grams	0.003	0.126	0.006

Table D.6: Clustering results for user A and D

Featureset	Homogeneity	Completeness	V-measure
CharFeatures	0.021	0.029	0.024
WordFeatures	0.004	0.006	0.005
SyntacticFeatures	0.022	0.034	0.027
StructuralFeatures	0.001	0.015	0.002
ContentSpecific	0.003	0.105	0.005
Char n-grams	0.003	0.105	0.005
POS n-grams	0.003	0.127	0.006

Table D.7: Clustering results for user B and E

Featureset	Homogeneity	Completeness	V-measure
CharFeatures	0.005	0.006	0.005
WordFeatures	0.002	0.002	0.002
SyntacticFeatures	0.004	0.106	0.007
StructuralFeatures	0.017	0.037	0.024
ContentSpecific	0.0	0.001	0.001
Char n-grams	0.048	0.092	0.063
POS n-grams	0.004	0.106	0.007

Table D.8: Clustering results for user C and F

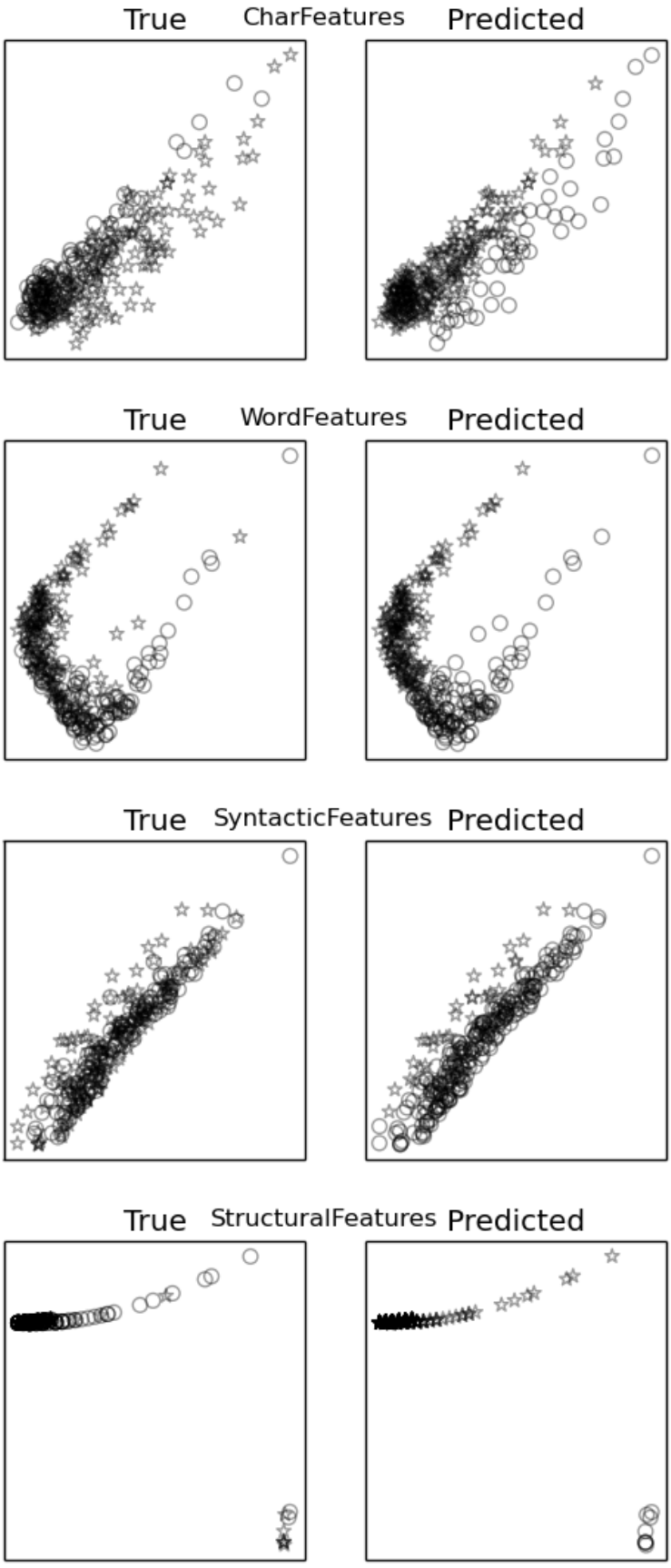


Figure D.4: Scatter plots for the clustering results using several feature sets for user A and D

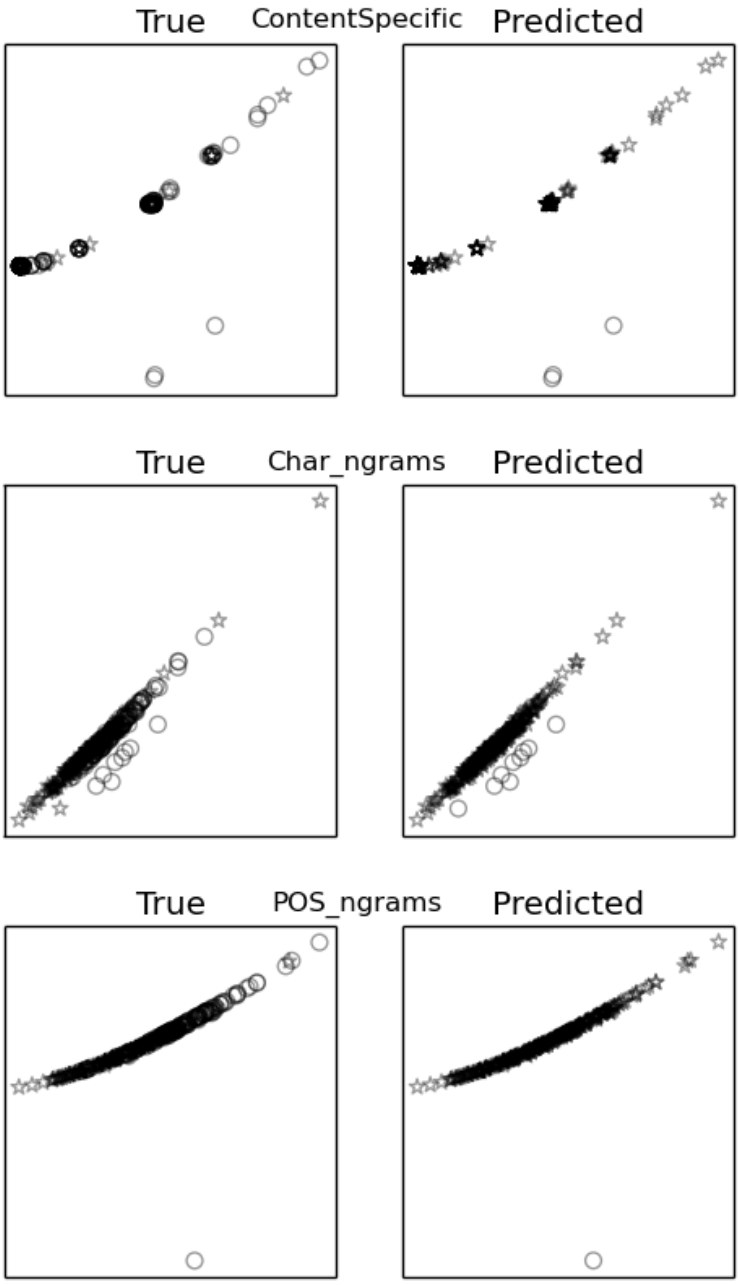


Figure D.5: Scatter plots for the clustering results using several feature sets for user A and D continued

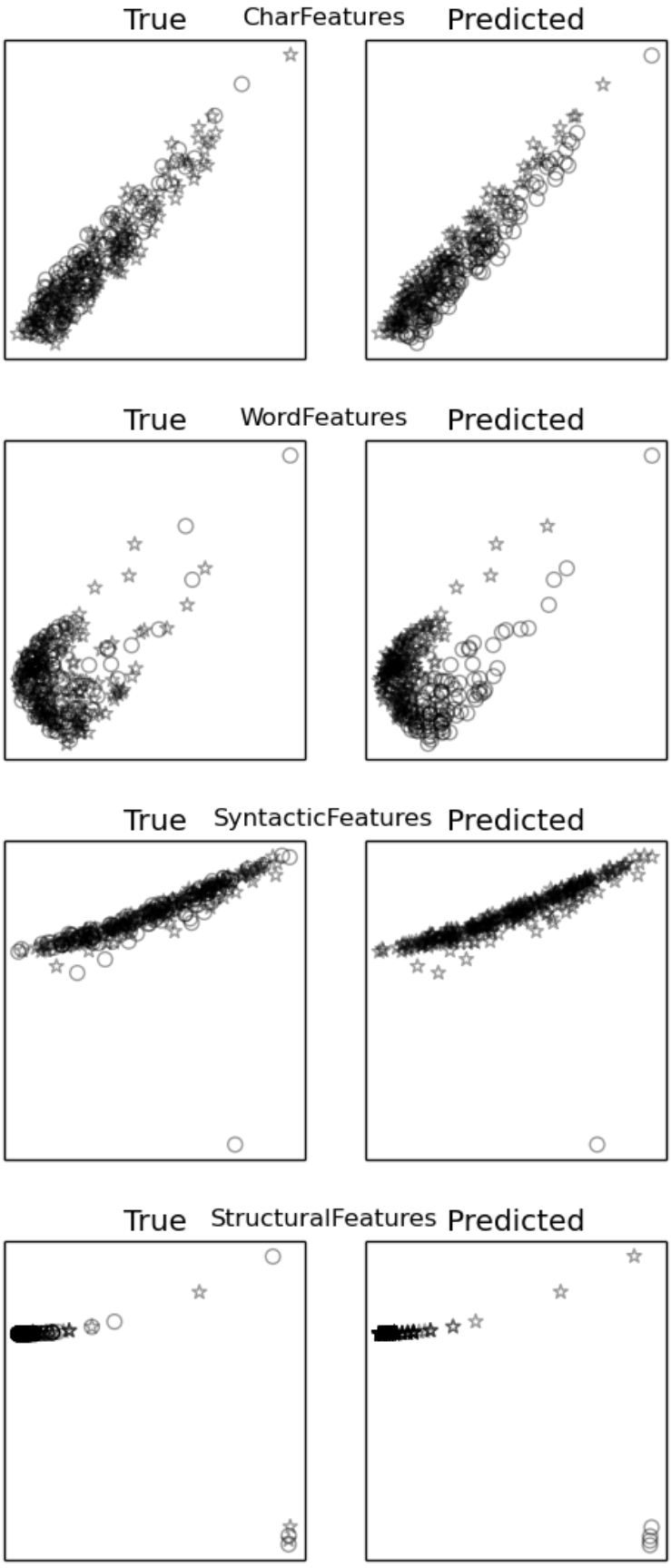


Figure D.6: Scatter plots for the clustering results using several feature sets for user B and E

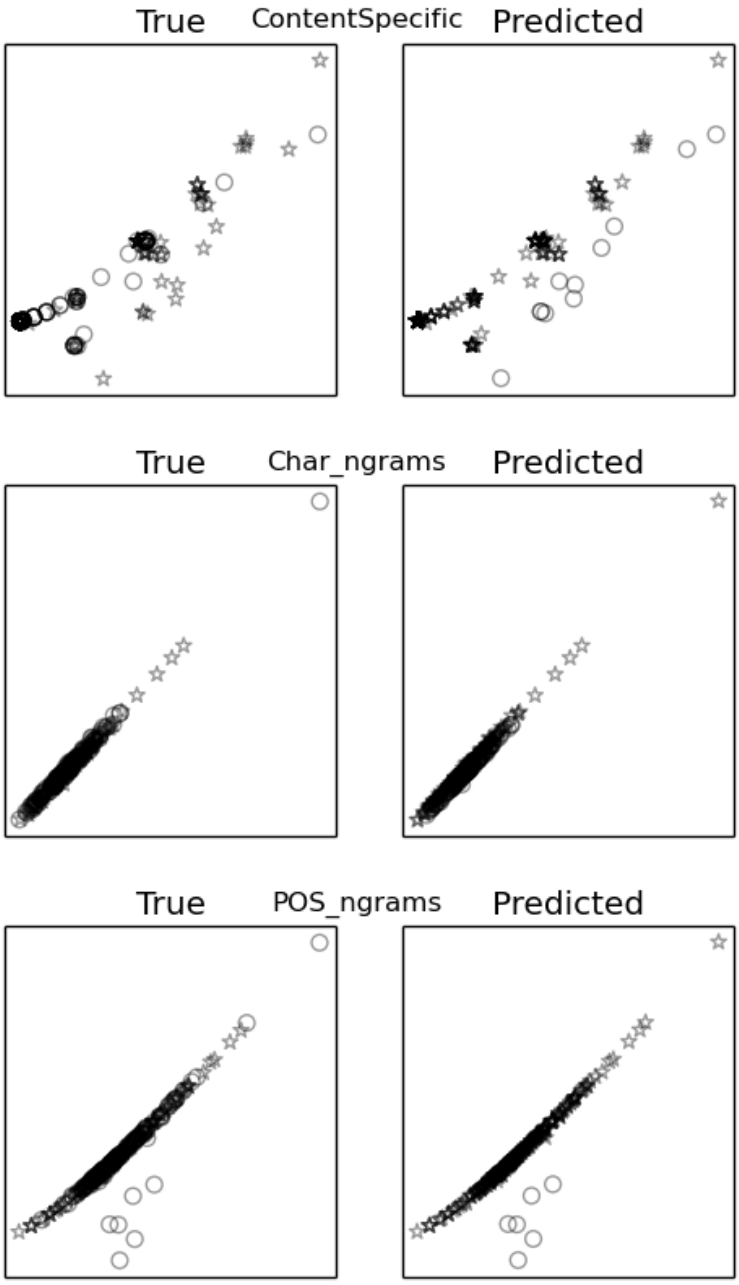


Figure D.7: Scatter plots for the clustering results using several feature sets for user B and E continued

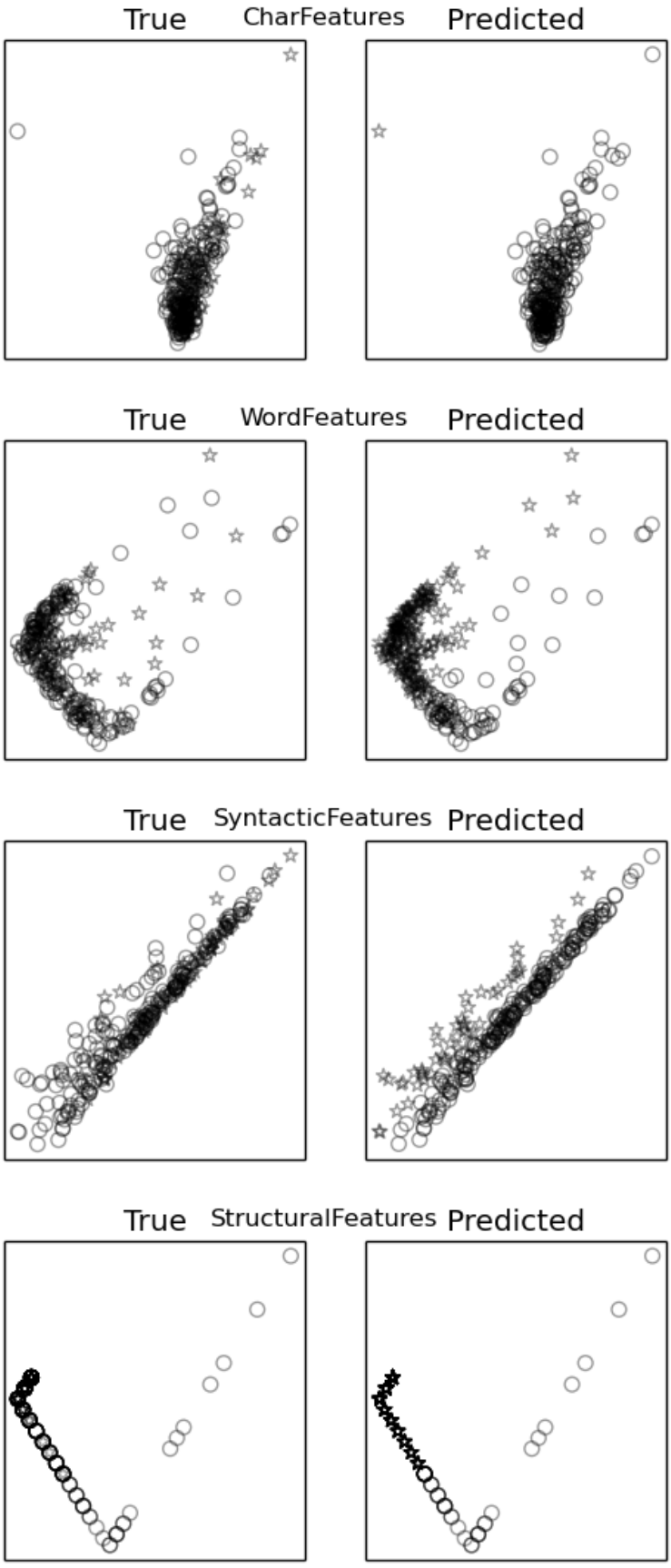


Figure D.8: Scatter plot for the clustering results using several feature sets for user C and F

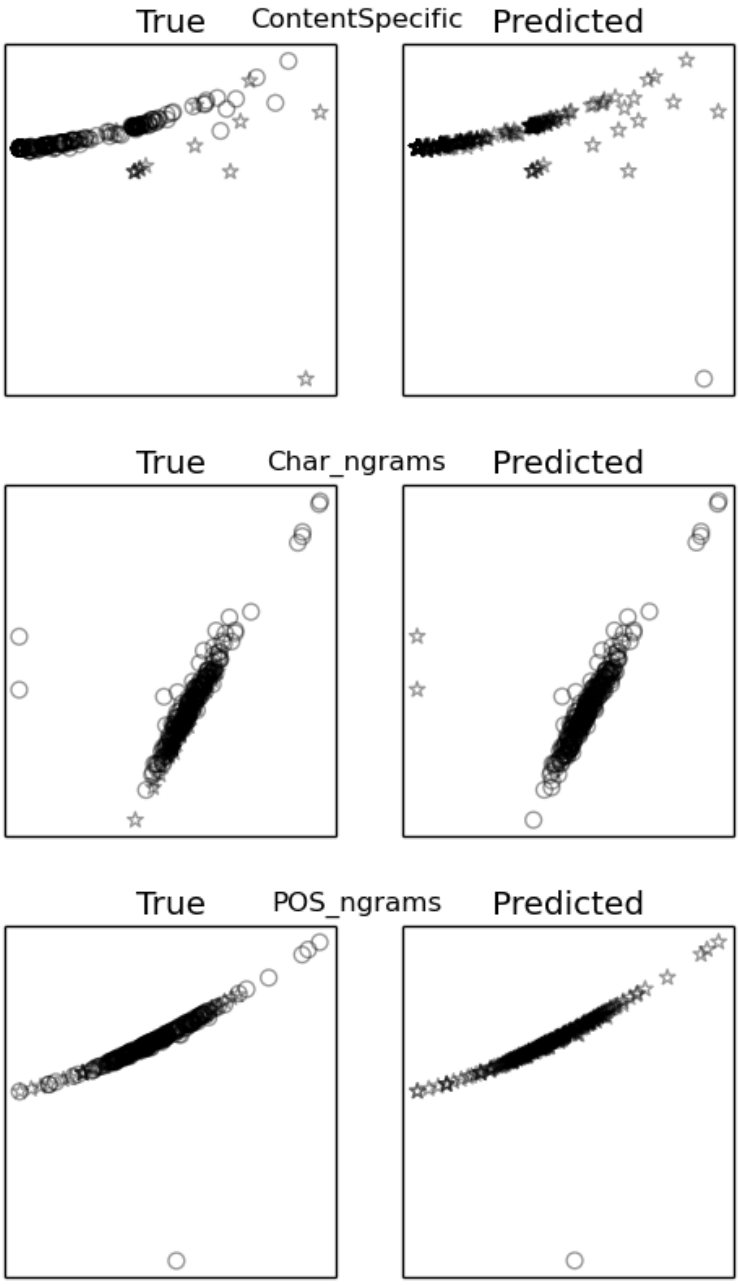


Figure D.9: Scatter plot for the clustering results using several feature sets for user C and F continued

D.2 Problem 1-2: Classification

D.2.1 Single feature sets

Features	P	(SD)	R	(SD)	F	(SD)
CharFeatures	0.75	(0.04)	0.83	(0.05)	0.78	(0.04)
WordFeatures	0.71***	(0.02)	0.84	(0.03)	0.73***	(0.03)
SyntacticFeatures	0.89***	(0.03)	0.86*	(0.04)	0.87***	(0.04)
StructuralFeatures	0.65***	(0.03)	0.77***	(0.04)	0.63***	(0.05)
ContentSpecific	0.84***	(0.05)	0.86*	(0.05)	0.85***	(0.04)
Char n-grams	0.96***	(0.03)	0.84	(0.03)	0.89***	(0.03)
POS n-grams	0.82***	(0.05)	0.67***	(0.04)	0.71***	(0.05)
Additional	0.62***	(0.01)	0.69***	(0.02)	0.48***	(0.02)

Table D.9: Performance of the classifier usig a linear kernel (*$p < .05$, **$p < .01$, ***$p < .001$)

Features	P (SD)	R (SD)	F (SD)			
CharFeatures	0.65	(0.03)	0.77	(0.05)	0.62	(0.06)
WordFeatures	0.67*	(0.02)	0.80**	(0.03)	0.66*	(0.03)
SyntacticFeatures	0.72***	(0.06)	0.81*	(0.05)	0.74***	(0.06)
StructuralFeatures	0.65	(0.03)	0.77	(0.04)	0.63	(0.05)
ContentSpecific	0.65	(0.03)	0.77	(0.05)	0.61	(0.09)
Char n-grams	0.61***	(0.02)	0.62***	(0.01)	0.62	(0.02)
POS n-grams	0.60***	(0.01)	0.61***	(0.01)	0.60	(0.01)
Additional	0.62***	(0.01)	0.69***	(0.02)	0.48***	(0.02)

Table D.10: Performance of the classifier using an RBF kernel (*$p < .05$, **$p < .01$, ***$p < .001$)

Class	P (SD)	R (SD)	F (SD)
Different	0.95 (0.02)	0.87 (0.03)	0.91 (0.02)
Same	0.54 (0.06)	0.79 (0.10)	0.64 (0.06)
Average	0.75 (0.04)	0.83 (0.07)	0.78 (0.04)

Table D.11: Results for CharFeatures - baseline

Class	P (SD)	R (SD)	F (SD)
Different	0.98 (0.01)	0.77 (0.03)	0.86 (0.02)
Same	0.44 (0.04)	0.91 (0.06)	0.59 (0.04)
Average	0.71 (0.03)	0.84 (0.05)	0.73 (0.03)

Table D.12: Results for WordFeatures

Class	P (SD)	R (SD)	F (SD)
Different	0.95 (0.01)	0.97 (0.01)	0.96 (0.01)
Same	0.84 (0.06)	0.74 (0.08)	0.78 (0.06)
Average	0.89 (0.04)	0.86 (0.04)	0.87 (0.04)

Table D.13: Results for SyntacticFeatures

Class	P (SD)	R (SD)	F (SD)
Different	0.97 (0.02)	0.64 (0.07)	0.77 (0.05)
Same	0.33 (0.04)	0.91 (0.06)	0.49 (0.05)
Average	0.65 (0.03)	0.77 (0.07)	0.63 (0.05)

Table D.14: Results for StructuralFeatures

Class	P (SD)	R (SD)	F (SD)
Different	0.96 (0.02)	0.94 (0.02)	0.95 (0.02)
Same	0.71 (0.08)	0.79 (0.09)	0.75 (0.07)
Average	0.84 (0.05)	0.86 (0.06)	0.85 (0.04)

Table D.15: Results for ContentSpecific

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.01)	0.97 (0.01)
Same	0.98 (0.04)	0.68 (0.06)	0.80 (0.05)
Average	0.96 (0.03)	0.84 (0.03)	0.89 (0.03)

Table D.16: Results for Char n-grams

Class	P (SD)	R (SD)	F (SD)
Different	0.89 (0.01)	0.98 (0.01)	0.93 (0.01)
Same	0.76 (0.08)	0.37 (0.08)	0.49 (0.09)
Average	0.82 (0.05)	0.67 (0.04)	0.71 (0.05)

Table D.17: Results for POSngrams

Class	P (SD)	R (SD)	F (SD)
Different	0.99 (0.01)	0.40 (0.03)	0.57 (0.03)
Same	0.25 (0.01)	0.99 (0.02)	0.40 (0.01)
Average	0.62 (0.01)	0.69 (0.02)	0.48 (0.02)

Table D.18: Results for Additional

D.2.2 Combinations of feature sets

Features	P (SD)	R (SD)	F (SD)			
Char n-grams	0.96	(0.01)	0.83	(0.02)	0.88	(0.02)
Ngrams+Char	0.97	(0.01)	0.83	(0.04)	0.88	(0.03)
Ngrams+Word	0.97	(0.01)	0.84	(0.02)	0.89	(0.02)
Ngrams+Synt	0.96	(0.02)	0.85	(0.03)	0.89	(0.02)
Ngrams+Stru	0.97	(0.01)	0.83	(0.02)	0.88	(0.02)
Ngrams+Cont	0.97	(0.02)	0.85	(0.03)	0.89	(0.03)
Ngrams+POS_	0.95**	(0.01)	0.81**	(0.02)	0.86***	(0.02)
Ngrams+Addi	0.96	(0.01)	0.82	(0.03)	0.87	(0.02)

Table D.19: Performance of the classifier using combinations of feature sets and a linear kernel (*$p < .05$, **$p < .01$, ***$p < .001$)

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.00)	0.97 (0.00)
Same	0.99 (0.02)	0.67 (0.04)	0.80 (0.03)
Average	0.96 (0.01)	0.83 (0.02)	0.88 (0.02)

Table D.20: Baseline: Results for Char n-grams

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.00)	0.97 (0.01)
Same	1.00 (0.01)	0.67 (0.07)	0.80 (0.05)
Average	0.97 (0.01)	0.83 (0.04)	0.88 (0.03)

Table D.21: Results for Ngrams+Char

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.00)	0.97 (0.00)
Same	1.00 (0.02)	0.68 (0.04)	0.81 (0.03)
Average	0.97 (0.01)	0.84 (0.02)	0.89 (0.02)

Table D.22: Results for Ngrams+Word

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.01)	0.97 (0.01)
Same	0.99 (0.04)	0.69 (0.06)	0.81 (0.04)
Average	0.96 (0.02)	0.85 (0.03)	0.89 (0.02)

Table D.23: Results for Ngrams+Synt

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.00)	0.97 (0.00)
Same	0.99 (0.02)	0.67 (0.05)	0.80 (0.03)
Average	0.97 (0.01)	0.83 (0.03)	0.88 (0.02)

Table D.24: Results for Ngrams+Stru

Class	P (SD)	R (SD)	F (SD)
Different	0.94 (0.01)	1.00 (0.00)	0.97 (0.01)
Same	0.99 (0.03)	0.69 (0.06)	0.81 (0.05)
Average	0.97 (0.02)	0.85 (0.03)	0.89 (0.03)

Table D.25: Results for Ngrams+Cont

Class	P (SD)	R (SD)	F (SD)
Different	0.93 (0.01)	1.00 (0.00)	0.96 (0.00)
Same	0.97 (0.02)	0.62 (0.05)	0.75 (0.04)
Average	0.95 (0.02)	0.81 (0.03)	0.86 (0.02)

Table D.26: Results for Ngrams+POS_

Class	P (SD)	R (SD)	F (SD)
Different	0.93 (0.01)	1.00 (0.00)	0.97 (0.00)
Same	0.98 (0.02)	0.65 (0.05)	0.78 (0.04)
Average	0.96 (0.01)	0.82 (0.03)	0.87 (0.02)

Table D.27: Results for Ngrams+Addi

Appendix E

Tables for problem 2

E.1 English

E.1.1 Single feature sets

Single messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.11	(0.16)	0.23	(0.18)	0.32	(0.18)
WordFeatures	0.07**	(0.15)	0.16***	(0.21)	0.23***	(0.23)
SyntacticFeatures	0.18***	(0.13)	0.34***	(0.15)	0.43***	(0.14)
StructuralFeatures	0.03***	(0.12)	0.09***	(0.19)	0.14***	(0.26)
ContentSpecific	0.10	(0.13)	0.18*	(0.15)	0.24**	(0.19)
Char n-grams	0.49***	(0.18)	0.65***	(0.17)	0.71***	(0.15)
POS n-grams	0.14	(0.10)	0.25	(0.13)	0.33	(0.14)
Additional	0.03***	(0.12)	0.07***	(0.22)	0.12***	(0.24)

Table E.1: Results for the English data set using single feature sets and single messages as input. SD represents the variation between users, *$p < .05$, **$p < .01$, ***$p < .001$)

Combinations of messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.11	(0.01)	0.24	(0.01)	0.32	(0.01)
WordFeatures	0.07***	(0.01)	0.16***	(0.00)	0.24***	(0.01)
SyntacticFeatures	0.18***	(0.01)	0.34***	(0.01)	0.43***	(0.01)
StructuralFeatures	0.03***	(0.00)	0.09***	(0.01)	0.14***	(0.01)
ContentSpecific	0.10***	(0.00)	0.19***	(0.00)	0.24***	(0.01)
Char n-grams	0.50***	(0.01)	0.66***	(0.01)	0.72***	(0.01)
POS n-grams	0.14***	(0.01)	0.25***	(0.01)	0.32	(0.01)
Additional	0.03***	(0.00)	0.07***	(0.00)	0.12***	(0.00)

Table E.2: Average results for the English data set after 5 runs, using single feature sets and single messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.20	(0.01)	0.37	(0.02)	0.47	(0.02)
WordFeatures	0.12***	(0.01)	0.27***	(0.01)	0.37***	(0.02)
SyntacticFeatures	0.31***	(0.02)	0.50***	(0.02)	0.60***	(0.02)
StructuralFeatures	0.03***	(0.01)	0.11***	(0.01)	0.17***	(0.01)
ContentSpecific	0.20	(0.02)	0.34***	(0.02)	0.42***	(0.01)
Char n-grams	0.70***	(0.02)	0.85***	(0.02)	0.89***	(0.02)
POS n-grams	0.22**	(0.02)	0.37	(0.02)	0.47	(0.02)
Additional	0.03***	(0.00)	0.09***	(0.00)	0.15***	(0.01)

Table E.3: Average results for the English data set after 5 runs, using single feature sets and combinations of 3 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.26	(0.02)	0.44	(0.03)	0.54	(0.03)
WordFeatures	0.15***	(0.01)	0.33***	(0.03)	0.44***	(0.02)
SyntacticFeatures	0.39***	(0.03)	0.58***	(0.03)	0.67***	(0.02)
StructuralFeatures	0.04***	(0.01)	0.13***	(0.01)	0.19***	(0.01)
ContentSpecific	0.27	(0.03)	0.44	(0.03)	0.53	(0.03)
Char n-grams	0.81***	(0.02)	0.92***	(0.01)	0.95***	(0.01)
POS n-grams	0.27	(0.02)	0.45	(0.02)	0.56*	(0.02)
Additional	0.03***	(0.01)	0.10***	(0.01)	0.16***	(0.01)

Table E.4: Average results for the English data set after 5 runs, using single feature sets and combinations of 3 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

E.1.2 Combinations of feature sets

Single messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
Char n-grams	0.50	(0.18)	0.66	(0.16)	0.72	(0.14)
Ngrams+Char	0.50*	(0.18)	0.66**	(0.16)	0.72	(0.14)
Ngrams+Word	0.50***	(0.18)	0.66**	(0.15)	0.73*	(0.13)
Ngrams+Synt	0.50***	(0.18)	0.66***	(0.16)	0.73**	(0.14)
Ngrams+Stru	0.50	(0.18)	0.66*	(0.16)	0.72	(0.14)
Ngrams+Cont	0.50	(0.18)	0.66	(0.16)	0.73	(0.14)
Ngrams+POS_	0.47***	(0.19)	0.64***	(0.17)	0.72	(0.15)
Ngrams+Addi	0.50	(0.18)	0.66**	(0.16)	0.72	(0.14)

Table E.5: Results for the English data set using combined feature sets and single messages as input. SD represents the variation between users, *$p < .05$, **$p < .01$, ***$p < .001$)

Combinations of messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
Char n-grams	0.50	(0.01)	0.66	(0.01)	0.72	(0.01)
Ngrams+Char	0.50***	(0.01)	0.66***	(0.01)	0.72**	(0.01)
Ngrams+Word	0.50***	(0.01)	0.66***	(0.01)	0.73***	(0.01)
Ngrams+Synt	0.50***	(0.01)	0.66***	(0.01)	0.73***	(0.01)
Ngrams+Stru	0.50***	(0.01)	0.66***	(0.01)	0.72	(0.01)
Ngrams+Cont	0.50**	(0.01)	0.66***	(0.01)	0.73**	(0.01)
Ngrams+POS_	0.47***	(0.01)	0.64***	(0.01)	0.72***	(0.01)
Ngrams+Addi	0.50**	(0.01)	0.66***	(0.01)	0.72***	(0.01)

Table E.6: Average results for the English data set after 5 runs, using combined feature sets and single messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.70	(0.02)	0.85	(0.01)	0.89	(0.01)
Ngrams+Char	0.71	(0.02)	0.85	(0.01)	0.89	(0.01)
Ngrams+Word	0.71	(0.02)	0.85	(0.02)	0.90	(0.01)
Ngrams+Synt	0.71	(0.02)	0.85	(0.02)	0.90	(0.01)
Ngrams+Stru	0.70	(0.02)	0.85	(0.01)	0.90	(0.01)
Ngrams+Cont	0.72*	(0.02)	0.86*	(0.02)	0.90	(0.01)
Ngrams+POS_	0.70	(0.02)	0.85	(0.02)	0.90	(0.01)
Ngrams+Addi	0.71	(0.02)	0.85	(0.01)	0.90	(0.01)

Table E.7: Average results for the English data set after 5 runs, using combined feature sets and combinations of 3 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
Char n-grams	0.81	(0.02)	0.92	(0.02)	0.95	(0.01)
Ngrams+Char	0.82	(0.02)	0.93	(0.01)	0.96*	(0.01)
Ngrams+Word	0.82	(0.02)	0.93	(0.01)	0.96*	(0.01)
Ngrams+Synt	0.81	(0.02)	0.92	(0.01)	0.95	(0.01)
Ngrams+Stru	0.81	(0.02)	0.92	(0.01)	0.95	(0.01)
Ngrams+Cont	0.82*	(0.02)	0.93	(0.01)	0.96**	(0.01)
Ngrams+POS_	0.81	(0.02)	0.92	(0.01)	0.96*	(0.01)
Ngrams+Addi	0.82	(0.02)	0.92	(0.01)	0.95	(0.01)

Table E.8: Average results for the English data set after 5 runs, using combined feature sets and combinations of 5 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

E.2 Dutch

E.2.1 Single feature sets

Single messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.19	(0.16)	0.38	(0.16)	0.49	(0.16)
WordFeatures	0.11*	(0.16)	0.28*	(0.22)	0.42	(0.23)
SyntacticFeatures	0.21	(0.13)	0.38	(0.13)	0.50	(0.13)
StructuralFeatures	0.08*	(0.20)	0.22*	(0.29)	0.32*	(0.32)
ContentSpecific	0.13	(0.09)	0.28	(0.21)	0.36*	(0.21)
Char n-grams	0.40***	(0.18)	0.60***	(0.16)	0.69***	(0.14)
POS n-grams	0.14	(0.10)	0.29**	(0.14)	0.40**	(0.15)
Additional	0.07*	(0.24)	0.18**	(0.29)	0.29*	(0.34)

Table E.9: Results for the Dutch data set using single feature sets and single messages as input. SD represents the variation between users, *$p < .05$, **$p < .01$, ***$p < .001$)

Combinations of messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.18	(0.02)	0.37	(0.04)	0.50	(0.04)
WordFeatures	0.12***	(0.02)	0.29***	(0.01)	0.41***	(0.02)
SyntacticFeatures	0.18	(0.01)	0.37	(0.02)	0.50	(0.01)
StructuralFeatures	0.08***	(0.01)	0.22***	(0.01)	0.31***	(0.02)
ContentSpecific	0.14***	(0.02)	0.28***	(0.01)	0.37***	(0.01)
Char n-grams	0.41***	(0.01)	0.61***	(0.02)	0.71***	(0.02)
POS n-grams	0.14***	(0.02)	0.29***	(0.01)	0.39***	(0.03)
Additional	0.07***	(0.00)	0.18***	(0.02)	0.29***	(0.01)

Table E.10: Average results for the Dutch data set after 5 runs, using single feature sets and single messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.26	(0.05)	0.45	(0.04)	0.58	(0.04)
WordFeatures	0.16***	(0.04)	0.38***	(0.05)	0.53***	(0.05)
SyntacticFeatures	0.24	(0.05)	0.45	(0.06)	0.58	(0.06)
StructuralFeatures	0.11***	(0.03)	0.27***	(0.03)	0.40***	(0.04)
ContentSpecific	0.26	(0.05)	0.47	(0.05)	0.56	(0.06)
Char n-grams	0.57***	(0.04)	0.77***	(0.03)	0.85***	(0.03)
POS n-grams	0.19***	(0.03)	0.37***	(0.04)	0.48***	(0.05)
Additional	0.07***	(0.02)	0.22***	(0.03)	0.36***	(0.05)

Table E.11: Average results for the Dutch data set after 5 runs, using single feature sets and combinations of 3 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.31	(0.06)	0.50	(0.07)	0.63	(0.07)
WordFeatures	0.21***	(0.05)	0.46	(0.07)	0.59	(0.07)
SyntacticFeatures	0.29	(0.07)	0.49	(0.06)	0.62	(0.07)
StructuralFeatures	0.12***	(0.02)	0.31***	(0.05)	0.43***	(0.05)
ContentSpecific	0.29	(0.07)	0.55*	(0.09)	0.64	(0.09)
Char n-grams	0.65***	(0.05)	0.82***	(0.05)	0.89***	(0.05)
POS n-grams	0.22***	(0.07)	0.41***	(0.07)	0.54***	(0.06)
Additional	0.06***	(0.03)	0.21***	(0.05)	0.36***	(0.05)

Table E.12: Average results for the Dutch data set after 5 runs, using single feature sets and combinations of 5 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

E.2.2 Combinations of feature sets

Single messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
Char n-grams	0.41	(0.17)	0.61	(0.18)	0.71	(0.17)
Ngrams+Char	0.41	(0.18)	0.61	(0.18)	0.71	(0.16)
Ngrams+Word	0.41	(0.18)	0.61	(0.18)	0.71	(0.16)
Ngrams+Synt	0.41	(0.18)	0.62*	(0.19)	0.71	(0.18)
Ngrams+Stru	0.41	(0.18)	0.61	(0.18)	0.71	(0.16)
Ngrams+Cont	0.41	(0.17)	0.61	(0.18)	0.71	(0.17)
Ngrams+POS_	0.42	(0.18)	0.62	(0.18)	0.71	(0.17)
Ngrams+Addi	0.42	(0.17)	0.62	(0.17)	0.72*	(0.15)

Table E.13: Results for the Dutch data set using combined feature sets and single messages as input. SD represents the variation between users, *$p < .05$, **$p < .01$, ***$p < .001$

Combinations of messages

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.41	(0.01)	0.61	(0.02)	0.71	(0.02)
Ngrams+Char	0.41	(0.02)	0.61	(0.03)	0.71	(0.02)
Ngrams+Word	0.41	(0.01)	0.61	(0.02)	0.71	(0.02)
Ngrams+Synt	0.41	(0.01)	0.62***	(0.03)	0.71	(0.02)
Ngrams+Stru	0.41	(0.01)	0.61	(0.03)	0.71	(0.02)
Ngrams+Cont	0.41	(0.02)	0.61	(0.03)	0.71	(0.02)
Ngrams+POS_	0.42***	(0.01)	0.62	(0.01)	0.71	(0.02)
Ngrams+Addi	0.42***	(0.01)	0.62***	(0.03)	0.72***	(0.02)

Table E.14: Average results for the Dutch data set after 5 runs, using combined feature sets and single messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.55	(0.06)	0.78	(0.05)	0.85	(0.04)
Ngrams+Char	0.57	(0.06)	0.77	(0.05)	0.85	(0.03)
Ngrams+Word	0.58	(0.05)	0.79	(0.05)	0.86	(0.05)
Ngrams+Synt	0.55	(0.06)	0.76	(0.04)	0.85	(0.03)
Ngrams+Stru	0.56	(0.06)	0.77	(0.04)	0.85	(0.04)
Ngrams+Cont	0.56	(0.06)	0.77	(0.05)	0.84	(0.05)
Ngrams+POS_	0.56	(0.06)	0.75	(0.05)	0.83	(0.04)
Ngrams+Addi	0.56	(0.05)	0.77	(0.03)	0.84	(0.03)

Table E.15: Average results for the Dutch data set after 5 runs, using combined feature sets and combinations of 3 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$

	Top1		Top3		Top5	
Feature set	Mean	(SD)	Mean	(SD)	Mean	(SD)
CharFeatures	0.65	(0.06)	0.83	(0.05)	0.90	(0.04)
Ngrams+Char	0.65	(0.07)	0.83	(0.06)	0.89	(0.05)
Ngrams+Word	0.65	(0.06)	0.84	(0.06)	0.90	(0.05)
Ngrams+Synt	0.67	(0.06)	0.84	(0.05)	0.89	(0.04)
Ngrams+Stru	0.64	(0.07)	0.82	(0.05)	0.89	(0.05)
Ngrams+Cont	0.65	(0.05)	0.83	(0.06)	0.90	(0.05)
Ngrams+POS_	0.66	(0.08)	0.83	(0.07)	0.90	(0.05)
Ngrams+Addi	0.66	(0.07)	0.85	(0.04)	0.91	(0.04)

Table E.16: Average results for the Dutch data set after 5 runs, using combined feature sets and combinations of 5 messages as input. SD represents the variation between runs, *$p < .05$, **$p < .01$, ***$p < .001$