FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

# Using Natural Language Processing to Automate the Bechdel Test

DIPLOMARBEIT

zur Erlangung des akademischen Grades

**Diplom-Ingenieurin**

im Rahmen des Studiums

**Computational Intelligence**

eingereicht von

**Krista Westphal BSc**
Matrikelnummer 00825466

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Univ.Prof. Dr. Allan Hanbury

Wien, 15. Jänner 2018

Krista Westphal    Allan Hanbury

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

# Using Natural Language Processing to Automate the Bechdel Test

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

**Diplom-Ingenieurin**

in

**Computational Intelligence**

by

**Krista Westphal BSc**
Registration Number 00825466

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Univ.Prof. Dr. Allan Hanbury

Vienna, 15th January, 2018

Krista Westphal

Allan Hanbury

# Kurzfassung

Der “Bechdel Test” stellt drei Fragen: hat ein Film mindestens zwei benannte weibliche Charaktere, unterhalten sich diese irgendwann im Film und gibt es eine Konversation zwischen weiblichen Charaktere über etwas anderes als einen Mann? Wenn alle drei Fragen positiv beantwortet werden können, besteht der Film den “Bechdel Test”. Diese Diplomarbeit beschäftigt sich mit der Automatisierung des Bechdel Tests für Drehbücher und Romane. Die Automatisierung dieser Aufgabe macht Analysen in großen Maßstab möglich, die bisher nur sehr aufwendig mit manuellen Methoden möglich waren. Somit könnten zum Beispiel Trends über längere Zeiträume analysiert werden. Bisherige Forschung existiert für das Automatisieren des Bechdel Tests für Drehbücher. Diese Forschung diente als Basis für den hier beschriebenen Ansatz. Obwohl der Bechdel Test ursprünglich für Filme konzipiert worden ist, sind die Fragen genauso anwendbar für Romane. Es existiert, soweit uns bekannt ist, keine Forschung für Romane zu diesem Thema.

Für Drehbücher parsten wir zuerst den Text mit einem neuen regelbasierten Ansatz, der auf die notwendige spezielle Formatierung eines Drehbuches aufbaut. Danach identifizierten wir alle Charaktere mit sprechenden Rollen und mit einem neuen Algorithmus, der die öffentlichen Zensus Daten mit dem “Internet Movie Database” (IMDb) Eintrag für den Film verbindet, um jeden Charakter ein Geschlecht zuzuordnen. Wir verwendeten auch einen “Machine Learning” Ansatz um zu prognostizieren ob es zumindest eine Konversation zwischen weibliche Charaktere gegeben hat, wo ein Mann nicht das Thema war. Die erreichten Ergebnisse für Drehbücher waren vergleichbar zu bisherigen publizierten Forschungsergebnisse.

Romane benötigen, aufgrund der strukturellen Unterschiede zwischen die zwei Textarten, einen anderen Ansatz als Drehbücher. Für Romane verwendeten wir ein “Named-Entity Recognizer” und einen regelbasierten Algorithmus um unterschiedliche Namen, die aber zum gleichen Charakter gehörten, miteinander zu verbinden. Die für Romane entwickelte Methode erreichte perfekte Genauigkeit bei einem kleinen Datenbestand von fünf Romanen.

# Abstract

The Bechdel test asks three questions: does a movie contain two named female characters, do two female characters converse at some point during the movie and is there at least one conversation between female characters that is not about a man? If all questions can be answered positively, then the film passes the Bechdel test. This thesis defines and implements methods for automating the Bechdel test for screenplays and novels. Being able to automate this task would allow for large-scale analyses, permitting researchers to analyse trends over long time periods, for example, that would otherwise only be possible with time consuming manual methods. Previous research exists for automating the Bechdel test for screenplays, which provided the basis for the approach described in this thesis. Although the Bechdel test was originally formulated for movies, the questions are just as applicable to novels. However, as far as we could find, no previous research exists for automating the Bechdel test for novels.

For screenplays we first parsed the text using a new rule-based approach that relies on the specialized text formatting required for screenplays. Then we identified all the characters who appeared in speaking roles and assigned each a gender by using a newly developed algorithm that incorporates census data about names and the Internet Movie Database (IMDb) information about the specific film. We also used a machine learning approach to predict if there is at least one conversation about something other than a man between the identified female characters. The results achieved for screenplays are comparable to the previous published work.

Novels required a different approach than screenplays, due to the differences in structure between the two texts. For novels we used a Named-Entity Recognizer and a rule-based algorithm that connects the different names used for each character throughout the text, to identify all the characters in a novel. Using quote attribution, we then determined which character says which lines of dialogue, and so establish who converses with whom. The method developed for novels achieved perfect accuracy on a small dataset of five novels.

# Contents

**Kurzfassung** **v**

**Abstract** **vii**

**Contents** **ix**

**1 Introduction** **1**

**2 Related Literature** **5**
2.1 Text Analysis . . . . . . . . . . 5
2.1.1 Natural Language Processing . . . . . . . . . . 6
Text Classification . . . . . . . . . . 7
2.1.2 Entities and their relations . . . . . . . . . . 8
Named-Entity Recognition . . . . . . . . . . 8
Gender Identification . . . . . . . . . . 9
Social Network Analysis . . . . . . . . . . 10
2.1.3 Quoted Speech and Attribution . . . . . . . . . . 11
2.2 Screenplay and Novel Analysis . . . . . . . . . . 13
2.2.1 Screenplay Analysis . . . . . . . . . . 13
2.2.2 Novel Analysis . . . . . . . . . . 15
2.2.3 Bechdel Test . . . . . . . . . . 15

**3 Methodology for Screenplays** **17**
3.1 Data . . . . . . . . . . 18
3.2 Structure of Screenplays . . . . . . . . . . 19
3.3 Parsing of Screenplays . . . . . . . . . . 21
3.4 T1: Are there at least two named female characters? . . . . . . . . . . 26
3.5 T2: Do at least two female characters talk to each other? . . . . . . . . . . 30
3.6 T3: Is there at least one conversation between them about something other than a man? . . . . . . . . . . 32
3.6.1 Features . . . . . . . . . . 34
3.7 Final Approach Integration . . . . . . . . . . 35

**4 Methodology for Novels** **37**

4.1 Data . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37
4.2 Format of a Novel . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
4.3 Parsing of Novel . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
4.4 T1: Are there at least two named female characters? . . . . . . . . . . . . 41
4.5 T2: Do at least two female characters talk to each other? . . . . . . . . . 44
4.6 T3: Is there at least one conversation between them about something other than a man? . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48
4.7 Final Approach Integration . . . . . . . . . . . . . . . . . . . . . . . . 49

**5 Results** **51**
5.1 General Bechdel Results for Films . . . . . . . . . . . . . . . . . . . . . 51
5.2 Results for Screenplays . . . . . . . . . . . . . . . . . . . . . . . . . . 52
5.3 Results for Novels . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 57

**6 Conclusion** **61**

**Bibliography** **63**

CHAPTER 1

# Introduction

In 1985 the cartoonist Alison Bechdel published a comic strip entitled “The Rule” for her “Dykes to Watch Out For” series where a character says she only goes to see a film “...if it satisfies three basic requirements. One, it has to have at least two women in it who, two, talk to each other about, three, something besides a man” [1]. This thesis uses the version of the Bechdel test with the additional requirement that to fulfill the first criterion the female characters must be named. The three requirements mentioned in the comic strip have become known as the “Bechdel Test”. The Bechdel test has proven to be a thought-provoking and much discussed test for films, novels, plays and more recently a more general test for male gender bias.

The test has been gaining in importance over the last 30 years and has become an integrated part of the zeitgeist. When a blockbuster movie is released, it is often accompanied by an article discussing whether the film passes the Bechdel Test or not [2]. In several Swedish movie theatres, films that pass the Bechdel test are marked with an “A” rating [3].

At first glance the requirements do not seem to set the bar very high. Therefore it is often surprising which films pass and which fail, as the test does not seem to require much; simply one conversation between female characters that is about something other than a man. Some films obviously pass, some films obviously fail, however a surprising number of films pass based on one or two lines of dialogue. If the viewer is not watching the film closely these could go by unnoticed.

Inspired by the original test, activists and journalists have recently started designing multifaceted variations of the Bechdel test [4] [5]. These variations often have to do with replacing “women” with another under-represented group, like minority characters. The test is also constantly being applied in new ways, in a variety of fields. For example the Bechdel test has recently been applied to social media platforms like Twitter [6].

Of course the test has its limitations. The idea behind the test is if a film is able to pass the Bechdel test, there is at least one important female character in the film. However there are films that pass based on a single line of dialogue in a single scene and there are films that don't pass, even though they have a strong female character.

For example, the German film *Run Lola Run* (1998) fails the Bechdel test, because no two named female characters converse during the course of the movie. The film *Captain America: Civil War* (2016) meanwhile passes on the basis of a single exchange of dialogue between the characters Scarlet Witch and Black Widow. Scarlet Witch says about a car parked on the street "You mean the red one? It's cute." and Black Widow replies "It's also bullet proof,...". There are films where it can be argued that female characters would be out of place. The movie *The Shawshank Redemption* (1994), which takes place almost exclusively in a mid-20th century male prison, is not going to have a lot of female roles. With that said, as an overall indicator, aggregating the results and not focusing on specific films, it has become a legitimate metric of male-bias in the film industry.

Having a method to automatically calculate if a screenplay or novel passes the Bechdel test would help large-scale analyses. A reliable algorithm that is able to calculate important features gives researchers the ability to focus on analysing and testing hypotheses, instead of watching thousands of movies or reading through thousands of novels.

The main concerns of this thesis are the interesting natural language processing, machine learning and linguistic problems an automatization of the Bechdel test provides. We define ways to automatically test if a film or novel passes the Bechdel test. For this we look at film screenplays and novels separately, as the algorithms needed to test each requirement vary significantly between the two formats. Each requirement allows us to explore different areas of natural language processing and text analysis. Since there has been very little previous research on automating this particular test, we rely on previous research focusing on the sub-problems posed by the test and insights on analysing these particular types of documents.

Starting with analysing screenplays, there are many interesting aspects and sub-problems to consider. A screenplay is an unusually well-structured text, as we will see in Chapter 3. The required formal formatting allows for us to have clean scene boundaries and to have a clear idea of who says which lines of dialogue. A novel does not have a required text structure, so therefore different strategies have to be applied. Nevertheless none of the requirements posed by the Bechdel test are trivial to calculate. Each has non-intuitive elements that have to be considered when designing algorithms that deliver good results.

Analysing novels allows us to focus on a much more freely structured type of text. As we will show, each requirement is much more difficult to automate with novels than with screenplays. Where with a screenplay we can parse each line and assign it a category that directly aids in answering some aspect of the Bechdel test, this is not possible with novels. We must apply more advanced techniques using limited annotated data.

We are not interested in measuring anything other than if a film is able to fulfill the three requirements or not. All of the focus of the features and structure of the algorithms deals explicitly with solving this specific problem. The focus is not on judging how effective the test is in measuring any kind of bias or interpreting any kind of additional meaning into the three requirements. We are focusing on the natural language processing tasks. The following list shows the main contributions made by this thesis:

- Definition of new rule-based approach for parsing screenplays
- Definition of new algorithm for determining gender of character in screenplay
- Definition of a method for evaluating whether a novel passes the Bechdel test

In Chapter 2 previous literature that is relevant to this thesis is presented. Chapter 3 focuses on the screenplay and discusses the methodology of automating the Bechdel test for screenplays. In Chapter 4 we discuss the methodology for literary texts. In Chapter 5 we evaluate the accuracy of our developed algorithms.

CHAPTER 2

# Related Literature

Finding a computational method to automate the Bechdel test for screenplays and novels means being able to process long written texts and obtain a limited understanding of the aspects within the text relevant for the Bechdel test. This starts with the basics, what is a word and progresses to, is this word a name? Is this sentence part of a dialogue and if yes, who is talking to whom? What are they talking about? These have been some of the fundamental questions of natural language processing (NLP) since its beginning and our task is to find a computational method that is able to answer all three Bechdel test requirements.

Automating the Bechdel test involves several different research areas. This section provides an overview of the most relevant. We look at NLP and some of it's most common applications. Research about social networks is relevant to the Bechdel test, as is automating name detection and gender identification. Additionally we provide a brief overview of previous research that uses the same types of texts, screenplays and novels, that we use in this thesis. Then we look at previous research done about the Bechdel test from a technical viewpoint as well as from a more humanities based viewpoint.

## 2.1 Text Analysis

The field of text analysis deals with the use of computational methods to extract and process data found in a wide variety of texts. In this section we describe the basics of some of the methods and applications that are relevant to the work in this thesis. First we discuss NLP and the task of text classification. We then look at entities and tasks relevant for this thesis, like gender identification. Lastly we consider the task of quoted speech attribution.

### 2.1.1 Natural Language Processing

Natural Language Processing (NLP) is a research area defined broadly as the study and developments of techniques that allow computers to understand and manipulate natural language [7]. NLP is very challenging as language is often ambiguous; it changes over time and comes in an ever increasing variety of forms, like courtroom transcripts, blog entries, news broadcasts etc. Language plays a central role in our lives and teaching a computer to understand it is a very complex task.

NLP has it's origins in the very beginning of machine learning and has consistently been an active research field since the first computers. In 1950 Alan Turing published the paper "Computing Machinery and Intelligence" where he introduced the test known today as the "Turing Test" [8]. This test continues to inspire many researchers. The goal of passing the Turing Test is to build a system so capable of understanding and responding to humans, that the human tester is unable to tell if their conversation partner is a computer or a human. For this to be possible, a machine must be able to understand the nuances of human language and also be able to produce language.

In the beginning NLP research mostly focused on token-based approaches [7]. A token breaks a text down into smaller components. Usually tokens are the words making up the text, however a token could also be a single letter, a phrase or even a whole sentence. This means researchers defined the rules that a language follows, the grammar and formal logic, and then applied this rule-based structure to texts. The more complex the problem is, the more difficult it is to define a comprehensive set of rules that can be applied to a wide variety of situations.

For example, words exist that have more than one meaning. One illustration of this is the word "note". Do we mean a note passed under the table between two school children or a note on a sheet of music? The computer trying to process the token "note" would need to have an extensive set of rules programmed into it, to be able to determine which meaning is the correct one. Many times we humans don't even notice ambiguities, since we have so much experience with them and use our knowledge and experience to intuitively make sense of them. Computers need to be taught these rules. Not to mention the fact that humans often ignore the rules governing their language, either on purpose or simply because they are unaware of them. How is a computer to know if an illogical use of a word is on purpose or not?

As researchers experimented with token-based approaches they realized that in order to create a computational method that can solve highly complex NLP problems, they might be better suited using a statistical, machine learning approach. In other words, using a large corpus of text to train a model to properly complete its language task. When creating a model to automatically translate between English and German, rather than programming all of the nuances and grammar of both languages in all levels of detail, define a large number of useful features, create a large corpus of already translated texts and train for example neural networks on the translations.

Deep learning (DL) is growing in prominence in the field of NLP [9]. Deep learning thrives on large datasets and due to the changing landscape of written content from a curated selection of published documents to large amounts of user-created content, the amount of available data is growing faster than ever [10]. One famous example of an application of DL is Google Translate. For years a phrase-based translation system was implemented for various language combinations. In 2016 the team switched to a DL algorithm for some language combinations and the accuracy improved significantly [11]. For the tasks being analysed by this thesis however, there currently isn't enough available annotated data to make DL a viable approach.

A common theme throughout NLP research is the importance of robustness. Texts coming from different sources are often formatted differently and these differences add an additional level of complexity when performing large-scale analyses. To achieve good results in cases like this, the training set needs to include a wide variety of texts. It is not sufficient to perform well on perfectly formatted texts. The algorithm must be able to process texts with anomalies as well.

### Text Classification

A classic application of NLP is text classification. The purpose of text classification is to assign a text to a predefined category. For example designing a spam filter is a text classification problem [12]. Does the email belong in the user's inbox or in the spam folder? The categories spam and non-spam are predefined and using features calculated from analysing the emails, an algorithm or model is able to determine a probability for which category is most likely the correct one.

One of the most commonly used strategies in NLP is called bag-of-words (BOW). When using this approach, all the words that appear in a text are viewed independently, without their surrounding structure. The result is a list of all the words that appear in the text with either a Boolean value or their multiplicity. If the goal is to compare several documents with each other, the total vocabulary of all documents is compiled and then the frequencies are calculated. Say a text contains 3000 different words, then 3000 features are calculated where the value is either a Boolean or the term frequency in that text.

BOW is helpful when comparing texts using keywords. Models created to work as spam filters use BOW features as an important component, although in sophisticated models always in combination with additional features. By ignoring the context, important information is neglected and therefore it is best to use a wide variety of features. That way context is still considered in some form.

One set of features that consider context to a certain extent are part-of-speech (POS) features. POS features involve tagging each text element with the part of speech, like noun, preposition, verb etc. When identifying which POS to assign to a token, the context and its surroundings play an important role, as does morphology [13]. POS tagging provides us with features that are used often in NLP models.

Another text classification feature is term frequency-inverse document frequency (TF-IDF). The idea is that when determining what themes one document in a collection of documents has, the words in the collection that appear infrequently, but are unusual, are the best basis. These unusual-to-the-collection words receive a proportionally higher score than words that appear often in many of the texts in the collection. Stop words are usually excluded, as they are not relevant in helping to determine what the text is about. A stop word is a word that commonly occurs, like "the" or "a". It is common practice to filter out these words before an analysis. The results are also normalized by the total number of terms that appear in the document so that long and short documents can be examined in the same collection. The determined topics of the document are then the $n$ words with the highest scores.

### 2.1.2 Entities and their relations

In this section we describe three areas of text analysis that deal with different kinds of entities. First we describe Named-Entity Recognition, where the goal is to find named entities of specific categories. For a certain text analysis application we might want to find all of the names or locations mentioned within a text. The second type of entity analysis we look at is gender identification, where we identify a named entity either as male or female. The third application we consider is social network analysis. After we have identified named entities, the goal is to determine who has some kind of interaction with each other.

#### Named-Entity Recognition

Before we can answer the first Bechdel test question "Do at least two named female characters appear?", we must be able to identify characters. This is best done by identifying names and Named-Entity Recognition (NER) can help us do this. NER takes a plain text and tags the named entities, according to pre-defined categories. For example the sentence "Christian went to the Eiffel Tower for the first time in 1990." would be tagged in the following manner:

"[Christian]$_{\text{Person}}$ went to the [Eiffel Tower]$_{\text{Location}}$ for the first time in [1990]$_{\text{Time}}$."

The research in this area started out in the Information Extraction field in the early 1990's and has become an established and mature area of research. As with NLP, in the beginning rule-based approaches were common. Now machine learning is playing a greater role, although the lack of annotated data in some domains means that rule-based strategies are still superior. Common features used for detection and classification include surrounding words, orthographic structure, prefixes, suffixes etc.

NER is usually broken down into two components. First is the detection of the names and second is then classifying the names found in the first step. For the detection and classification many techniques have been applied and [14] provides an excellent overview of the most commonly used ones. Classification is sometimes done with a specific domain in mind, although general approaches using machine learning also exist and are becoming more popular. We used the most well-known general NER tagger, the Stanford NER tagger [15], as the first step when identifying characters in a novel. In related character identification research this approach of using a modified NER was also taken [16] [17].

### Gender Identification

As the amount of available unstructured data has increased, interest has grown in finding an efficient way to assign a gender to a specific name. Having additional metadata information allows for researchers to conduct broad analyses comparing users with different traits. The data could prove to be even more useful if researchers knew more about the people writing the texts. For example when researching gender bias in journalism, it is necessary to have the gender information of the journalists as well as the people who appear in the articles.

There are several methods for gaining gender information. The easiest is simply to ask a user to self-identify. Facebook is an example of a social media platform that requires users to provide gender information. However most platforms do not require users to self-identify and research has shown that when supplying gender information is not mandatory, the information is often not provided voluntarily. Women are even less likely to voluntarily provide gender information [18] and this has to be accounted for when performing any analyses that use opt-in gender information.

Another gender identification method is to look at a name manually and do some additional background checks to identify the person's gender. This has been used on analyses pertaining to Twitter, as a profile picture is often attached to a Twitter profile [19]. Researchers only used profiles with a profile picture of an actual person and discarded users with non-photograph profile pictures. Each images was looked over by five different people, who classified each user profile as either female, male or unknown.

Both of these methods, self-identification and manual identification, are non-applicable for our purposes. We need to find a method that is scalable and is able to automatically determine gender, without any manual input, using only a name.

Previous attempts of automatically assigning gender have focused mostly on a dictionary approach. This means compiling a reference database and comparing the name in question with the data entries in the database to find the gender. The reference databases are often some form of publicly available census data. Applying this method is fairly straightforward, however there are several weaknesses to this approach. The gender of a name is often not as unambiguous as perhaps first assumed. The US Census data from

1890 to 2010 contains thousands of distinct names that appear as both female and male names. Another challenge is different cultures have different naming customs. For the method to be as broadly applicable as possible, it is necessary that non-English names are also correctly assigned genders. Not all countries make their census data publicly available and even if they did, it would be a lot of work to find and normalize all of the different datasets.

In some cases it is also possible to infer gender based on the content. Gendered pronouns that appear in a text often signal the subject's gender. A more speculative method is to use language style [20]. This approach could be used when attempting to assign genders to authors of novels or users of Twitter [21]. This type of research is similar to the problem described in Section 2.2 of determining the authorship of a text. The text itself is used to find distinguishing features and the result is a classification of the author's gender.

Privacy and ethics are also concerns when dealing with collected data belonging to real people. Since our analysis only deals with characters from screenplays and novels, protecting user privacy is not an issue.

**Social Network Analysis**

The second question of the Bechdel test is "Do two female characters talk to each other" at any point in the film. Who is interacting with whom or what can be asked about a wide variety of data, not just screenplays. This type of analysis is called social network analysis (SNA) and we were able to use findings from this field to help us answer this second requirement.

In our case we want to determine who interacts with whom during each scene of the film. Other applications of SNA include analysing Twitter data to find out which users interact with each other and how they are connected [22]. Social media companies in general use SNA, as it allows them to use their data to add value for a user. When Facebook suggests potential friends, it is using SNA to find people outside of your current network, who share many similarities with others who are already in your network. Another intuitive use of SNA is in the analysis of diseases and how pandemics occur [23].

SNA represents social interactions using networks and graph theory. The nodes are individuals and the edges connect individuals who have some kind of interaction. What that interaction is, is defined depending on what question is being asked of the data. The interaction could be talking to each other, a retweet on Twitter or simply physically being in the same room together. Based on the structure of the resulting network, clusters can be identified and patterns made visible that might otherwise have gone undetected.

After a graph has been created, a wide variety of features can be calculated based on the structure of the nodes and edges. If we want to find the most central characters, we are looking for the nodes with the largest number of edges, known as the "degree centrality". Another common measurement is the density, or how well connected the nodes are. This is often useful in comparing networks with each other. Which features to calculate depends on what question is being asked of the data.

One of the first examples of formal SNA research was in the 1930's by the psychiatrist Jacob Moreno, although informal interest in social networks has existed for centuries. Moreno looked at social networks in a prison setting [24] [25] and using questionnaires filled out by the inmates, he was able to describe and analyse the networks within the prison. He used the resulting network to visually show the subjective feelings the inmates had for each other. He attempted to visually represent in one graph who interacts with whom the most.

Another application is identifying people of interest, based on the large amount of metadata being collected by various agencies all around the world. An especially extreme example is terrorism. Governments around the world are using SNA principles to try to identify potential terrorists [26]. Since the amount of information collected by a government is enormous, the process of using this collected data to find suspected terrorists must be as automatic as possible. It would also be best to have as few false positives as possible, so the analysis has to be very reliable.

### 2.1.3 Quoted Speech and Attribution

When reading a novel, the reader intuitively knows who is saying which lines of dialogue. Even if there is no direct attribution within the immediate text, we are able to deduce who the speaker of each line of dialogue is using our knowledge of written dialogue conventions. One of the major challenges of this thesis is designing a program that enables a computer to also be able to determine exactly who said which line of dialogue.

There has been some research done in the area of journalism and identifying quote attribution within news articles [27]. Reporters use two main types of reported speech. *Direct speech*, indicated with quotation marks around what the source said and *indirect speech*, where the reporter summarizes what was said and does not use quotation marks. However journalism is a simplified case study compared to novels. The techniques used for quote attribution in the news domain, when applied directly to literary texts, do not perform well [28].

The research in the area of quote attribution for literary texts is fairly limited. Before any quantitative research can be done, annotated datasets need to be created. The creation of such datasets is an extremely time intensive task. Elson et al. [29] annotated 60 novels using Amazon's Mechanical Turk service. Later researchers have found fault with these annotations however, as due to their quality restrictions many quotes in their dataset are not attributed to a speaker. The corpus created for [30] covers only three novels, however each novel has been annotated manually by expert annotators. Two of the works from this corpus, *Pride and Prejudice* and *Emma* from Jane Austen, are included in our dataset. We did not include the third work, Anton Chekhov's *The Steppe*, since it was not originally written in English, but translated from Russian.

The first step of quote attribution in a literary text is determining who the characters are. Who are the possible characters that we can draw from? Sometimes this step is done manually or simply assumed to be available. Other times, as in our case, there is a preparatory step where this master list is created.

An author has several tools at their disposal for telling the reader who said what. One common device is "coreference", sometimes referred to as anaphora. To be able to decipher coreferences, we need to look at the surrounding text. For example when looking at this excerpt from *Emma*:

> **Frank Churchill** turned towards her to listen.
> "You were speaking," said **he**, gravely.

Here "he" refers to the previously mentioned character "Frank Churchill". Another tool authors use commonly is the dialogue chain. A dialogue chain is when the same character has several sequences of quotes, but only the first quote is attributed explicitly. This quote from *Pride and Prejudice* is an example of a dialogue chain:

> "No really," replied Elizabeth; "I think there cannot be too little said on the subject."

The first part is directly attributed to the character Elizabeth. The second quote is also said by her, but not attributed to her directly.

The three approaches we drew on from the literature are all similar in that they use textual cues to categorize quotes, however the methodology of how to go from the category to the actual speaker varies. The approach described in [28] assigns one of seven categories to each quote found in the text, based on syntax. Using features calculated from the syntax of the sentence containing the quote as well as features from surrounding text for each candidate character, predictive models are created and evaluated. They achieved 83% accuracy of speaker selection using a dataset composed of manually identified quotations from 19th and 20th century literature. The method described in [31] achieved a speaker identification accuracy rate of 79% by using a rule-based approach. The focus is on finding the speech-verb, from there finding the Actor and then to find the Speaker. The dataset utilized was a corpus of manually annotated books of fiction, of unknown composition. Muzny et al. [30] created an annotated corpus that supports their quote attribution method of a deterministic sieve-based approach, achieving an average accuracy of 78% on a dataset consisting of three manually annotated novels. In this context, the idea of a sieve-based approach is to create a list of rules where the text is evaluated in the order that the rules are listed. If a match is found for the first rule, then it is evaluated according to the first rule. If there is no match, then the text is evaluated according to the second rule and so on.

## 2.2 Screenplay and Novel Analysis

When analysing texts like screenplays and novels, we aim to combine computational methods with research ideas out of the humanities, which is often called digital humanities. As the capabilities of computational methods have grown from word analyses to machine learning models, many humanities researchers have started using NLP methods to test hypotheses they have. There have been some errors along the way. The most famous was when Shakespeare scholar Donald Foster attributed a text to Shakespeare using an author attribution model, which turned out to be inaccurate [32]. This failure created a lot of scepticism and serves as a warning for the careful application these methods.

The movement towards the digitalization of literature has made large-scale studies of texts possible [33]. Before so many texts were available online or in databases, linguists and other scientists interested in language practised “close reading” analysis. A very intense analysis of a small number of texts. With the advances of the last few years a new approach can be used: distant reading. Distant reading allows researchers to constantly consider the big picture. What other texts were published in the same time frame and how do writing styles or subject matter compare [34]? A researcher doesn’t even have to necessarily read the works they are studying, just program a computer to extract the information needed.

The labeling of the Federalist Papers using text analysis methods is an interesting example of NLP being able to solve a long standing debate among researchers [35]. The Federalist Papers are 85 essays written primarily by Alexander Hamilton and James Madison prior to the ratification of the United States constitution, all published under the pseudonym “Publius”. Approximately 12 essays could not be conclusively assigned an author, since we have no written record from the time of who wrote which essay. This lack of proof of authorship has caused many debates among historians.

Researchers trained a model based on known writings of the two men using vocabulary and writing style features. Then they ran the trained model on the disputed federalist papers. The result was a probability of authorship and the result has become widely accepted in the field of American History.

Below we discuss research pertaining specifically to screenplays and novels, the two types of text that are used in this thesis. We also define the Bechdel test and examine previous work on the Bechdel test.

### 2.2.1 Screenplay Analysis

The first type of text we examined in this thesis are screenplays, since movies are the original application area of the Bechdel test. Computational methods allow for a large scale analysis of screenplays, to recognize trends that might otherwise be hard to detect. The screenplays available online, spread out over various websites, provide the basis for almost all analyses. There is no accepted corpus of screenplays, however there are corpora for movie dialogue and movie summaries that have been the basis for several research papers [36].

Numerous books have been written about how to best tell a story in the screenplay format [37] [38]. Some focus on profitability, some on the fundamentals of movie storytelling and still others on formatting. These books are usually anecdotal and rarely use a data-based approach or research.

Promising work has been done in the area of visualizing the social network within a screenplay and showing when which characters interacted with each other [39]. Using line diagrams, where a line represents each main character, the program bunches and crosses lines to show when which characters were together in the same scene or at the same place.

One study examined what makes a line in a screenplay memorable [40]. Why do certain phrases stick in our minds, while others are quickly forgotten? The model was trained using an IMDb database of memorable quotes, looking to be able to identify the lines of dialogue that would stick. They found memorable quotes distinguished themselves by being general and distinctive. Generality was measured by the use of personal pronouns, indefinite articles and fewer past tense verbs. To measure distinctiveness they compared the words in the memorable quotes with a database of journalistic texts. The more distinctive the words found in the quote were, the more likely it was to be memorable.

Using deep learning and a large number of science fiction screenplays, a related experiment generated a completely machine written screenplay [41]. It was then produced and can be viewed online. The authors struggled with many of the same issues of this thesis, like normalizing a large number of screenplays with various different formats. Most of their corpus comes from TV shows like *X-Files* and *Star Trek*, which traditionally have a different story arch than a film. Automatically writing movie screenplays is still in the beginning stages and a lot of progress has to be made before a machine written script would be seriously produced and released.

Another line of research attempts to quantify what makes a screenplay successful in order to better automatize the process of deciding which screenplays to make into films [42]. Here a wide variety of metadata about the screenplay is collected and features are created that are then compared to films that have already been produced and released. Also basic features about how expensive it will be to make the movie are calculated in order to predict the probability of a high return on investment (ROI). It may be cynical, but ROI plays a decisive role when studios decide which films to produce. Similarity measurements between films with a high ROI and the screenplay in question are calculated. Then the model decides which screenplays are most likely to have a high ROI.

### 2.2.2 Novel Analysis

Although the Bechdel test was not originally formulated to test novels, the requirements can be applied to literary texts just as well as to screenplays. Many more novels have been written and published than full-length movies released, because the format has existed for longer and the threshold for creating a final product is lower. Thanks to initiatives from private and public institutes to scan literary works and make them available to the public, many published novels are now available to us in some type of electronic form. For example Google has scanned over 25 million texts for their project Google Books [43].

Research using novels has attempted to answer a wide variety of literary questions. Sometimes the focus is on a specific literary work and other times the goal was a broad analysis, using as many novels as the researchers could find. A well-known example of an examination of a single work is Wallace et al. [44]. The research here pertains to the over 1,000 pages long novel *Infinite Jest* by David Foster Wallace. The novel has three main storylines that alternate in irregular intervals throughout the novel. The goal was to automatically separate out each storyline in order to be able to make sense of the novel's structure.

One example of a broad analysis used approximately 1,300 literary novels to try to find similarities in their story arcs [45]. The researchers extracted reader sentiments, related to sentiment analysis, from the texts in their sample and were able to identify six patterns that fit almost all of the texts. They concluded that although plots vary greatly between novels, the number of types of overreaching story arcs are limited to six.

A book written by Franco Moretti also looked at a wide variety of European novels and identified locations that were mentioned within the text [46]. These locations were then manually assessed by experts, to consolidate and condense the list of locations, perhaps also considering historical changes that have taken place. The location names were then placed on a map and various comparisons were made, like visualising all of the locations that appear in Jane Austen's novels. Doing a task like this manually is very costly. With this kind of additional geolocation information, large-scale analyses can use this addition feature for testing hypotheses proposed by literary analysts.

### 2.2.3 Bechdel Test

For a film or novel to pass the Bechdel test, the following three sub-tests must be passed:

- T1: Are there at least two named female characters?
- T2: Do these female characters have a conversation with one another?
- T3: Is there at least one conversation between female characters about something other than a man?

Failing one of the tests means that the text has failed the Bechdel test. If the screenplay or novel satisfies all criteria in each of the three sub-tests, then it passes the Bechdel test.

Agarwal et al. [47] defined a method of automating the Bechdel test for screenplays and the paper is closely related to the work presented in this thesis. Their research developed algorithms for solving each element of the Bechdel test separately, using rule-based techniques for solving T1 and T2 and machine learning for T3. Their approach for solving T1 is similar to what we implemented for this thesis, however in Agarwal et al. [47] they used scene descriptions as well as the character name to help determine the gender of the names that appear in the screenplay. Our method uses only the character's name. We followed the same approach for T2, but for implementing T3 we added a new category of features and removed features that did not perform well. We obtained the data used in their research and used their results as comparison in Section 5.2.

In the social sciences field there has been research done in the area of gender bias and movies [48]. Garcia et al. [6] uses computational methods to measure the Bechdel test on the social media platforms MySpace and Twitter. They created a character network based on the movie screenplay and used Twitter and MySpace data to connect conversations about movie trailers to posts on these platforms.

Another possible method for calculating the Bechdel test is to use actual video and audio of a film, instead of the screenplay. This would require different skills from the method described in this thesis, however recent research has been promising. Guha et al. [49] developed an automated system to analyse the amount of screen time male and female characters have, with the goal of creating a gender representation indicator for a specific movie. It would be feasible to adapt this kind of work to look specifically for the information needed to automate Bechdel test.

CHAPTER 3

# Methodology for Screenplays

The screenplay is the foundation of the film-making process. It fulfills many different roles for a diverse set of people. A producer looks at the screenplay to estimate how many locations will be necessary and how much the project will cost, among other things. An actor uses the script to study the scenes they will appear in and how they might best deliver their lines. A cinematographer sorts out which types of cameras will be needed for each scene and how they might be best positioned. All of this information needs to be packed into a well-formatted document, where each party is able to then select what information is relevant for them.

This is also true when examining a screenplay to test if it passes the Bechdel test. As is the case with the film crew, not every line is relevant when addressing the question of whether a film passes or fails, so we need to carefully consider the document to pick and choose the information that is needed. Each part of the test focuses on a different set of elements within the screenplay.

As a consequence of irregular formatting often found in screenplays, a very robust preprocessing is necessary before any of the algorithms for T1, T2 or T3 can be applied. The preprocessing used here annotates each line of text with a label. These labels are described in Section 3.2 and are necessary for determining which lines need to be looked at closer for each part of the Bechdel test. After the preprocessing is complete, the screenplay is run through the algorithms for T1, then T2 and finally T3.

The algorithms contain a mix of rule-based and machine learning components. Since a screenplay has a comparatively formal structure, well designed rule-based approaches can achieve good results. However NLP has made a lot of progress in applying machine learning approaches and they are the most cutting-edge model for many applications.

This chapter of the thesis will summarize the general structure of a screenplay and the methodology of automating each test for movie screenplays. One difficultly of using screenplays to conclude if a film passes the Bechdel test or not, is that what appears on the page is not necessarily what appears on the screen. Last minute changes, improvisations or deletion of scenes in the editing room can lead to irregularities. However the assumption made here is that the changes are rarely so significant as to change the result of any of the Bechdel tests.

## 3.1 Data

For several types of written texts, like newspaper articles or speeches, there are well known corpora that contain a representative, well-balanced selection of texts of a specific category that meet certain criteria. Unfortunately there is no such widely accepted corpus for screenplays, although there is a website dedicated to curating screenplays. The website is called the “Internet Movie Script Database” (IMSDb) [50]. The database is very comprehensive, although the quality in formatting varies from screenplay to screenplay. This database proved to be the best basis for our analysis and has also been used in the relevant previous literature [51] [52].

A screenplay usually contains between 6,000 and 8,000 lines of text. The long length of a single screenplay makes a manual analysis of a large corpus problematic. Previous research that analyses screenplays usually crawl the scripts from websites that publish the scripts embedded in websites or as PDF files. We have avoided PDF files, because it is notoriously difficult to convert them into plain-text with the correct formatting.

As mentioned above the screenplay is written before the final film has been released, usually before filming starts. Sometimes it is edited afterwards to reflect any changes that were made during filming and editing, but there is no guarantee that this has occurred. Within the framework of this thesis it is not feasible to analyse these discrepancies, so we have decided to use the scripts as they are.

In testing our results we used the dataset from [47]. This dataset includes many screenplays with a diverse range of structures from many genres. Since our goal is to create a strategy that is as widely applicable as possible, we want to test it on as many different screenplays as possible. Specialising for a certain type or only being able to process perfectly formatted scripts is not our goal. The developed strategy should be able to work with a wide variety of data.

Now that we have found usable screenplays, how do we identify the films that actually pass the Bechdel test? Fortunately there is a very comprehensive, crowd-sourced resource online[1]. The crowd-sourcing aspect is important, because it allows anyone to update any entry. Answering the question, does a film pass the Bechdel test, is typically

[1] www.bechdeltest.com

easy to do after having watched a film carefully, with the criteria in mind. However in some cases just one line of dialogue makes the difference and these cases can cause some intense discussion about whether the film should receive a passing grade or not. There is daily activity on the website, so we can reasonably assume that it is up-to-date.

Since it would not be feasible in the time available to watch all of the films being analysed, the results found on this site are our ground truth. Each film being analysed in our database has an entry on the website. Exporting the results is very convenient as an API is provided. The result of the export is a data entry that contains the movie title, year of release, IMDb ID, Bechdel rating from 0 to 3 and a "dubious" Boolean value to describe any doubt in that rating expressed in the comments and quantified by the site masters. A Bechdel rating from 0 to 3 is sufficient to tell us which criteria the film fulfills, since the criteria are dependent on each other. A Bechdel rating of 2 means the film passes the first two criteria, but all of the conversations between the female characters are about men.

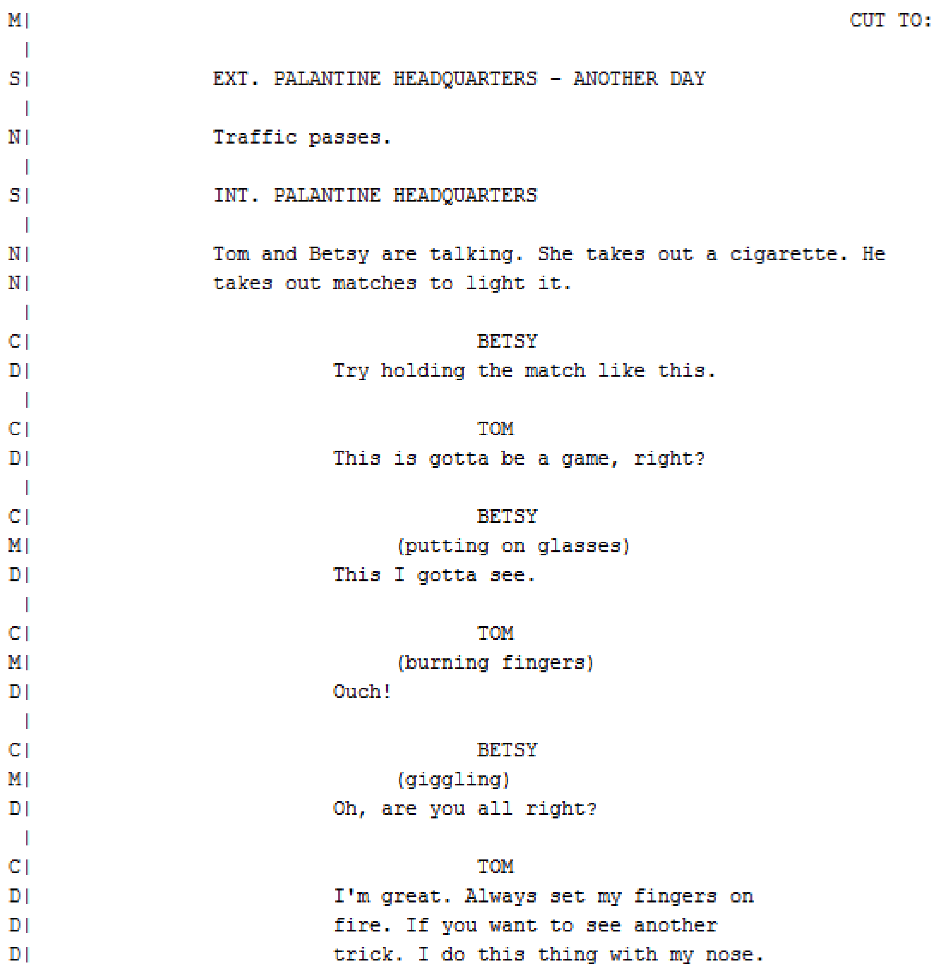
## 3.2 Structure of Screenplays

As explained in [53], "A filmscript is a semi-structured textual document in that it is subdivided into scenes and sometimes other structural units. Both character dialogue and also descriptive and background metadata is provided in a filmscript. The metadata is partially formalized." As a consequence of their semi-structured nature, screenplays are an excellent candidate for computational analysis. As we will see in Chapter 4, the analysis of more free-form texts like novels poses additional challenges.

The basic syntax and grammar used when formatting the screenplay is the basis for the parsing method described in Section 3.3. Ideally all screenplays would use the exact same syntax, but this is unrealistic. There are general agreed upon best practices, however these rules are not always followed and any parsing algorithm must be able to deal with anomalies. Although there are some well established best practices, many books have been written about screenwriting and numerous websites exist that are devoted to the topic of writing screenplays, screenplay authors do not always follow them or they invent their own rules [54] [55]. Additionally software exists that automatically formats a screenplay. The best known software and the industry standard is called "Final Draft".

The fundamental elements of a screenplay are best explained using an example. Figure 3.1 shows an excerpt from the screenplay of "Taxi Driver" (1976). This basic scene illustrates the most common elements found within a screenplay. The capital letter at the beginning of each line of text is not part of the original document, but the result of parsing. The original text is what appears after the '|' symbol.

When looking at the excerpt the first thing one notices are that formatting choices, especially capitalizations and indentations, are an important part of the screenplay's



```
M|                                                          CUT TO:
 |
S|             EXT. PALANTINE HEADQUARTERS - ANOTHER DAY
 |
N|             Traffic passes.
 |
S|             INT. PALANTINE HEADQUARTERS
 |
N|             Tom and Betsy are talking. She takes out a cigarette. He
N|             takes out matches to light it.
 |
C|                                  BETSY
D|                     Try holding the match like this.
 |
C|                                  TOM
D|                     This is gotta be a game, right?
 |
C|                                  BETSY
M|                          (putting on glasses)
D|                     This I gotta see.
 |
C|                                  TOM
M|                          (burning fingers)
D|                     Ouch!
 |
C|                                  BETSY
M|                          (giggling)
D|                     Oh, are you all right?
 |
C|                                  TOM
D|                     I'm great. Always set my fingers on
D|                     fire. If you want to see another
D|                     trick. I do this thing with my nose.
```

Figure 3.1: Excerpt from the screenplay of the film “Taxi Driver” (1976). Labels on the far left are result of parsing.

structure. A scene starts with a scene boundary, sometimes referred to as a slug line, labeled in Figure 3.1 with an “S”. The scene boundary gives the reader general information about where the scene will take place (INT. for interior, EXT. for exterior etc.), the name or description of the location (“PALANTINE HEADQUARTERS”) and the time frame or time of day (“ANOTHER DAY”). The scene boundary is capitalized and marks the beginning of a scene. A scene ends when the next scene boundary appears in the script.

To set the tone of the scene, often times the scene boundary is followed by a scene description, labeled with “N”. The scene description has the same indentation as the scene boundary and is used to convey additional background information. In our example there are two passages of scene description. One sets the mood for the building exterior; the second sets up the dialogue.

Then there is the interplay between character names, labeled with “C”, and dialogue lines, labeled with “D”. Character names are capitalized and indented so that they appear about in the middle of a page. Dialogue is indented further than scene boundaries, but not as far as the character names. The exact level of indentation varies a great deal from screenplay to screenplay, however the proportion of the indentation is usually consistent. The character name always comes before the dialogue and is sometimes even refined with additional descriptors like “V.O.” for “Voice-Over”, “O.S” for “Off-Screen” or “CONT’D” for “Continued”. This pattern of scene boundary, scene descriptions, character names and dialogue repeats itself for the full length of the screenplay.

Additionally a metadata category is necessary for additional lines that don’t fit in any of the four main categories. Two common types of metadata lines are present in the screenplay excerpt shown in Figure 3.1. The first type are camera cues like “CUT TO”, “CROSSFADE” etc. The second type are additional directions on delivery of dialogue, which are contained in parenthesis between “C” and “D” lines and are on a different indentation level than the true dialogue. These lines are labeled with an “M”. Lines without any text do not need a label and can be ignored.

## 3.3 Parsing of Screenplays

Before any detailed analysis of a screenplay is possible, the screenplay must be parsed. In our case, parsing a screenplay consists of assigning a label to each line of the screenplay. The assigned label will be used as a basis for determining if a screenplay passes T1, T2 and T3. The parsing method used should be robust enough to deal with anomalies that commonly occur in screenplay documents and also be able to recognize screenplays that are too irregular to be properly parsed.

There are two main paths for parsing that can be taken. The first is a rule-based algorithm and the second is using a machine learning approach. In Agarwal et al. [51] both a rule-based and machine learning approach were implemented. They concluded that the machine learning method delivered better results. However, after analysing the rule-based approach used, we found that important structural information like indentation had not been incorporated. Since we have so much information about the structure of screenplay documents, we can use this knowledge to design a robust rule-based parsing approach that performs comparably to the machine learning approach found in Agarwal et al. [51].

A basic element of our parsing algorithm is the indentation level. When measuring the indentation level of a line of text, we count the number of spaces that occur between the start of the line and the first element of text. After this feature has been calculated for each line in a screenplay, we analysed the frequencies of certain indentation levels. As an example, here is a graphic representation of the indentation levels in the screenplay for the movie *The Sixth Sense* (1999):

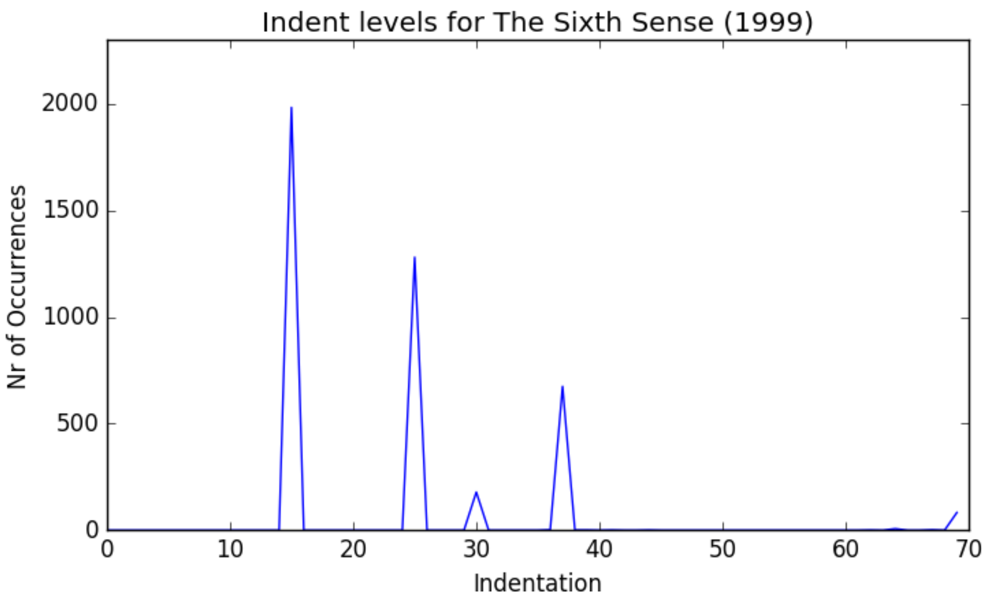


Figure 3.2: Indentation for screenplay *The Sixth Sense* (1999)

The graph shows that there are certain levels of indentation that appear very frequently and some that are not used at all. These peaks in the data represent the indentation level of commonly appearing elements. The first peak in Figure 3.2 is the indentation level of 15 spaces for the “S” and “N” lines and is also the smallest indentation that appears in the screenplay. We labeled this indentation level as “Indent Level = 1”. The second peak at 25 spaces is for the “D” lines and is “Indent Level = 2”. The third, much smaller peak, at 30 spaces is where the metadata describing the dialogue appears. In our example, 179 lines have "Indent Level = 3". The fourth peak "Indent Level = 4", with 37 spaces, is for the "C" lines. Then there is a very small peak at 70 spaces, also representing "M" data, like in our example excerpt shown in Figure 3.1 the line containing "CUT-TO". These lines have "Indent Level = 5". The Table 3.1 provides an overview of each label and the indent level associated with it.

Throughout the screenplay there are isolated scattered indentations, but in a well-formatted screenplay these should occur very rarely. We found in our research that the indentation levels usually follow the pattern of the example. The individual indentation levels vary widely, however these peaks are almost always visible.

Another feature that is relevant for the structure of a screenplay is capitalization. Certain types of lines are written in all-caps, like the "S" and "C" lines. The Table 3.1 shows which types of lines are usually capitalized and which are not.

An additional important deciding feature when determining the parsing label of a line in a screenplay is the presence of keywords. We manually compiled a dictionary of keywords for scene boundaries by looking through screenplay writing guides and documentation for the software program "Final Draft". Since these keywords are so distinctive and unlikely to appear otherwise, they are helpful when parsing the screenplay.

| Label | Indent Level | Capitalized | Keywords |
|---|---|---|---|
| S | 1 | Yes | Yes |
| N | 1 | No | No |
| C | 4 | Yes | No |
| D | 2 | No | No |
| M | 3/5 | Sometimes | Sometimes |

Table 3.1: Overview of parsing label and defining characteristics

Each line of the screenplay containing text is given one of five possible labels, described in the Section 3.2. Lines labeled with "S" mark when a scene begins. They are the basis for determining when a scene begins and are therefore vital for T2. "C" lines mark the names that need to be analysed and are the basis for T1 and for determining who is in a scene for T2. "D" lines mark the conversations that take place and also provide the information on what the characters are talking about, which is the basis for T3. Lines labeled with "N" and "M" are used by the screenwriter as ways to add descriptions and background information. We also use "N" lines to help improve our accuracy when testing T2.

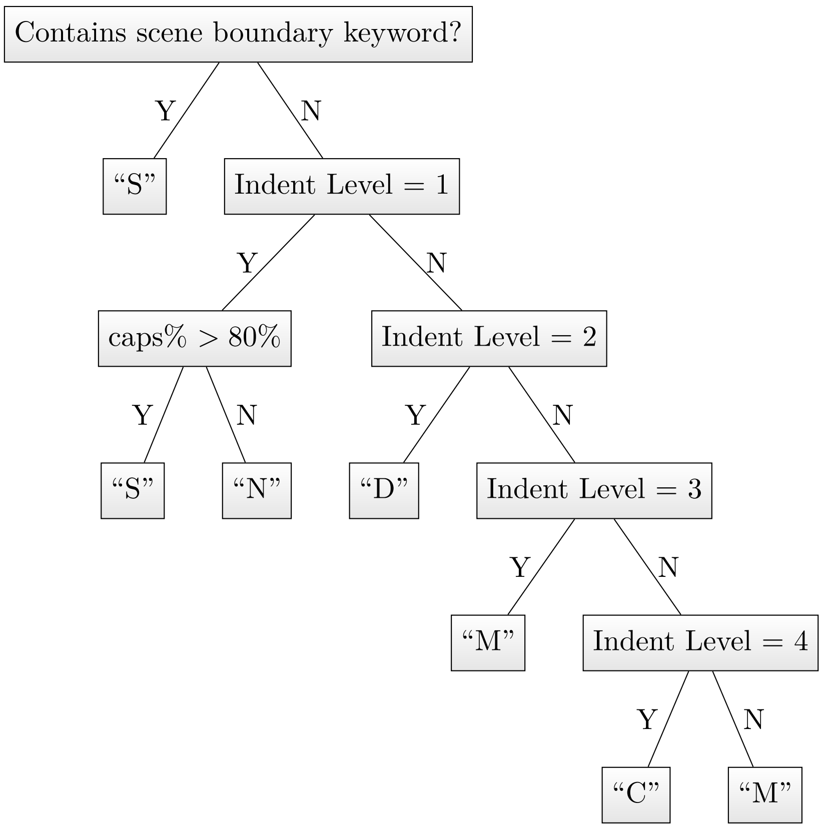


Figure 3.3: Decision tree for parsing screenplays

Each of these labels can be assigned based on just a few attributes when the screenplay complies with the best practices described in Section 3.2. The "C" lines have a unique indentation and are written in all-caps. The "D" lines also have a unique indentation and "S" lines start with just a handful of words that can be saved in a dictionary. Based on these requirements, three attributes are all we need when assigning a label to a line of text: indentation, % capitalized and whether the line contains a scene boundary keyword or not.

In the first pass each line is saved in an array along with the indentation, % capitalized and if keywords appear in the line. In the second pass the indentation levels are analysed and the hierarchy described above in Table 3.1 is established. We are looking for the four or five most common indentation levels. After we have determined the indentation hierarchy, we know that the indentation with the fewest number of spaces most likely belongs to "S" and "N" lines, the second indentation level belongs to the "D" lines and the fourth indentation level "C" lines. The other indentations most likely belong to "M" lines or are anomalies.

Based on the calculated features and level of indentation, the line is run through the

| Label | Possible Successive Labels |
|---|---|
| S | N,C,M |
| N | S,C,M |
| C | D,M |
| D | S,N,C,M |
| M | S,N,C,D |

Table 3.2: Possible order of parsing labels

decision tree shown in Figure 3.3. First we look-up if the line contains one of the following scene boundary keywords: “INT”,“EXT”, “INT/EXT” or “I/E”. These keywords are very distinctive and therefore a good indicator of a line that should be labeled with “S”. If no such keyword is found, we move on to the indent levels. Lines that have the first level of indentation are most likely either an “S” or “N” line. We differentiate between these two categories by the percentage of the line that is capitalized. If over 80% is capitalized, then the line is labeled with an “S” and for whatever reason does not contain a scene boundary keyword. Otherwise the line is labeled with an “N”. We progress through the indent levels, labeling each line accordingly. The lines with a level of indentation higher than 4 are labeled as “M”.

The parsing results are then run through two sanity checks. The first sanity check is the number of different indentation levels. If 90% of all lines fall into 3 indentation levels and there are no more than 6 different indentations with more than two lines, then this sanity check is passed. As a second sanity check, we check to see if the resulting structure is coherent and harmonious. A screenplay tells a story and has a logical structure. This means that “C” lines come before “D” lines, that “S” lines are often followed by “N” lines. To test this, we compressed all of the labels assigned during the parsing step into a list, eliminating successive duplicate entries. We then defined which labels can logically follow each other. Table 3.2 shows the labels that would logically follow the different labels. If we find more than 5% of the scenes have some form of illogical structure, then the screenplay fails the second sanity check.

The most commonly occurring deviation or anomaly are incorrect capitalizations. Either a line is capitalized that would not normally be, maybe to add extra emphasis, or a line that should be capitalized is written in lower-case. Our decision tree is able to deal with these anomalies, as relative indentation levels are the main criteria. All screenplays in the dataset from Agarwal at el. [47] pass the sanity checks defined here. This can be credited to the fact that the screenplays used were already found to be well-defined by their work.

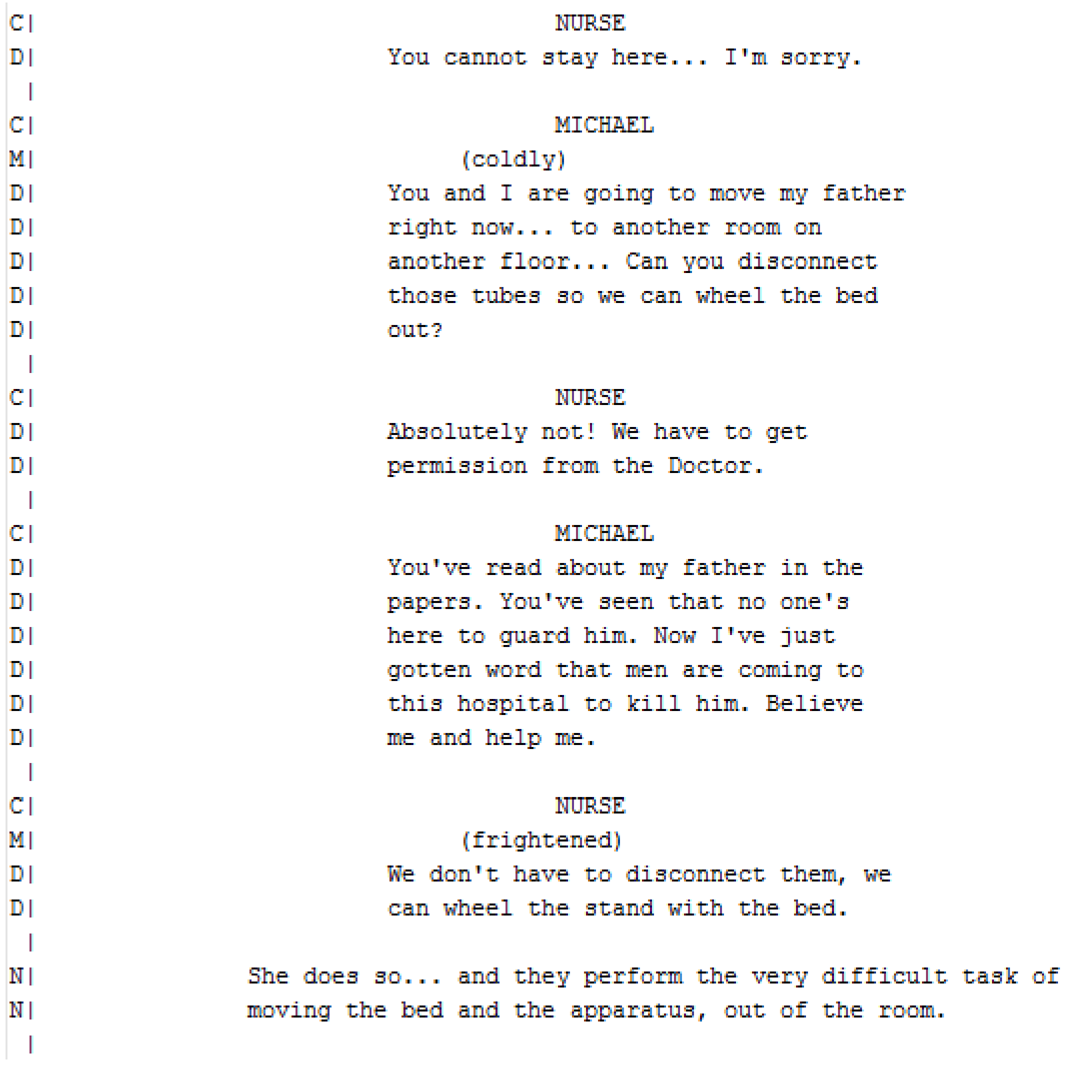
## 3.4 T1: Are there at least two named female characters?

For a screenplay to pass T1 it needs to contain at least two named female characters. From the parsing step we know which lines contain character names and can use the content of these lines to compile a list of suspected characters in a screenplay. This list is then run through an advanced name-to-gender algorithm, designed using elements first introduced in Agarwal et al. [47]. However their final approach used the context surrounding the name, by running a NER and coreference resolution on the scene descriptions and assigning the gender based on the most common gender of the third person pronouns. The approach described here uses only the text of the “C” lines, which eliminates the time-consuming task of running coreference resolution over parts of the screenplay.

As shown in Figure 3.4, the character name used in the screenplay is not always a real name, but more a descriptor. Some characters go unnamed and are referred to only by such descriptors. For example in the film *The Godfather* (1972), there is a scene where a nurse interacts with Michael, one of the main characters. In the screenplay Michael is named, however the nurse is simply referred to as “NURSE” and is not named at any point. When reading the script carefully it is often possible to nevertheless determine the gender of a character, because of the pronouns used in the scene descriptors. This is also the case in the excerpt shown in Figure 3.4. The reader knows the nurse is written as a female character, because the pronoun “she” is used in an “N” line. The person watching the film also sees that this character is female. For our purposes however, we cannot include these characters as named characters. Therefore they do not count towards fulfilling the criteria for T1, even if the character is female.

Another problem case is when the same character is referred to differently at various points in the screenplay. For simple variations in the names it is possible to design a similarity parameter to filter out those double names. However if a character is first referred to as “OFFICE WORKER” and is later referred to by a real name like “CLARA”, it is no longer possible, only using the “C” lines, to figure out that those are the same character. Using different names for the same character could also be an artistic device used by the screenwriter in order to add suspense. A screenplay is a written work and the screenplay writer wants the reader to stay engaged and continue reading. For example in *Fantastic Four* (2005), the screenplay refers to a character first as a “SHADOWED FIGURE” and after the character is revealed to the other characters, he is referred to by his actual name “VICTOR”. Writing the screenplay like this adds suspense, but does not make our task easier.

Even when a character is referred to consistently by the same name, the gender of the name could be ambiguous. In English it is possible that a name is unisex and therefore cannot easily be assigned to only one gender. An example of a unisex name is “Jamie”, which was a popular boy name and then later became a popular name for girls. There are many examples of these kinds of names, so our gender identification algorithm



```
C|                         NURSE
D|               You cannot stay here... I'm sorry.
 |
C|                         MICHAEL
M|                    (coldly)
D|               You and I are going to move my father
D|               right now... to another room on
D|               another floor... Can you disconnect
D|               those tubes so we can wheel the bed
D|               out?
 |
C|                         NURSE
D|               Absolutely not! We have to get
D|               permission from the Doctor.
 |
C|                         MICHAEL
D|               You've read about my father in the
D|               papers. You've seen that no one's
D|               here to guard him. Now I've just
D|               gotten word that men are coming to
D|               this hospital to kill him. Believe
D|               me and help me.
 |
C|                         NURSE
M|                    (frightened)
D|               We don't have to disconnect them, we
D|               can wheel the stand with the bed.
 |
N|          She does so... and they perform the very difficult task of
N|          moving the bed and the apparatus, out of the room.
 |
```

Figure 3.4: Excerpt from the screenplay of the film *The Godfather* (1972)

needs to be able to find the character's gender, even when the character's gender isn't clear based purely on a name analysis.

In designing a name-to-gender algorithm, we used two different approaches. Each approach has its strengths and each has its weaknesses. Using a combination of both methods proved to be the most successful strategy.

**Dictionary**: Using a compiled list of baby names and their popularity from the years 1890-2010 [56], we created a reference dictionary of names. Assigning a gender means finding the corresponding entry in the dictionary and allocating the gender of the name as the gender of the character. As a standalone strategy, this approach proved to be not very successful, because it is badly equipped to deal with the common anomalies described above that appear in screenplays.

The algorithm developed needs to be able to cope with several common irregularities. As mentioned above, when a character is referred to by their job title for example, then this character cannot be assigned a gender. When characters are referred to only by their last names, the dictionary approach is unable to find a sensible match. The compiled dictionaries consist of first names and it is not possible to determine gender based on a last name. This does not mean that there aren't screenplays where this strategy alone would be sufficient in correctly assigning genders. However our goal is to define an algorithm that is applicable to as many different screenplays as possible.

The focus of our work is on English screenplays, but that does not mean that non-English names do not appear as characters in the screenplays. Adding a similar baby name database for the UK [57] improved our results, however in most cases when the dictionary approach did not work because the name could not be found in our compiled dictionary, the IMDb technique worked much better.

**IMDb**: The website "www.imdb.com" (IMDb) offers an API that allows for users to retrieve character name, actor name and gender of actor for any film in it's comprehensive database. We used the Python package IMDbPY [58] to access the IMDb data. This approach is therefore based on matching the character name in the screenplay with a character name on the casting list.

There are some problems with combining the data retrieved from IMDb with the parsed "C" lines. The name that appears in the screenplay may not match the character name listed in IMDb. For this we developed a similarity parameter, so that if there was no perfect match we could find the closest match. We must filter out the non-named characters beforehand, since they are irrelevant. The IMDb data is especially helpful for characters that in the screenplay are not called by their first name. In those cases we have no first name to run through the dictionary, so we are reliant on matching the character name with the name found in the IMDb database.

**Algorithm 3.1:** T1: Assign gender to character names

```
Data: list of character names
Result: list of character names and corresponding gender
while nextEntry() $\neq$ null do
    name = nextEntry() ;
    name = removeDescriptors(name) ;
    if nameInDict(name) then
        if genderCertainty(name, 0.7) then
            name.gender = genderDict(name) ;
            break ;
        end
    end
    actor = findIMDB(name, 0.8) ;
    name.gender = actorGender(actor) ;
end
```

In the Algorithm 3.1 we show how a name is assigned a gender. First any additional descriptors are removed from the "C" line. As described in Section 3.2 sometimes keywords like "V.O." for "Voice-Over" occur in the same line as the character's name. This extra text needs to be removed, as it is not an actual part of the character's name. Since the dictionary approach is fast and effective for simple cases, this is used in the first step. A gender certainty feature is calculated for female and male by finding all instances of the name in the database and dividing the occurrences in the specific gender by the total number of occurrences. The resulting percentage is the gender certainty. For example in our database the name "Jamie" occurs 240 times, 128 times as a female name and 112 times as a male name. This means a gender certainty of 53% for female and 47% for male. Those values are below the gender certainty parameter of 70%, which is an adequate threshold for the corpus used in this thesis. That means that a character with the name "Jamie" would have to be assigned a gender using the IMDb method. Other names, like "Cornelia", have an almost 100% female gender certainty.

If the name could not be found or the gender certainty is below the value of the gender certainty parameter, the IMDb approach is used. Using the IMDb ID, we retrieve the list of characters that appear in that specific film along with the actors who played the roles and their gender. If the character name used in the screenplay matches a retrieved entry from IMDb or the character name is part of only a single name found in the list of characters for that film, then that actor's gender is assigned to the character. For example the character "Michael" appears in the excerpt shown in Figure 3.4. The IMDb entry for this character is "Michael Corleone". However, since there is only one character in the film named "Michael", a match is possible in this step.

If no match could be made, we used a syntactic similarity measurement to improve the match quality between the name in the "C" line and the name of the role found on IMDb. The Levenshtein distance is the measurement we used, which is defined as the minimal number of single-character edits required for one of the words to become identical with the other word [59]. The lower the Levenshtein distance, the more similar the words. In the Algorithm 3.1 this similarity measurement, transformed into a percent, is the second parameter in the findIMDB function. The percentage is calculated by dividing the Levenshtein distance by the maximum number of edits, which is equal to the length of the longer name. We found that names that are less than 80% similar are not accurate matches. If a match could be found, then we retrieved the gender of the actor who played the role and assigned that gender to the character.

If no match could be found using either method, a name is not assigned a gender. In our sample data this happened 12% of the time. If a gender cannot be assigned with a minimal amount of certainty, then the character gets assigned "U" for unknown gender. The dialogue entries of any character names labeled "U" do not play a role in answering T2 or T3.

Now that we have a gender for the "C" lines we can go through the list of characters and add-up the number of named female characters. If there is a total of more than two named female characters, the screenplay is passed onto the function for testing T2. If we were unable to find two named female characters, then the screenplay has failed the Bechdel test and receives a score of 0 from 3.

## 3.5 T2: Do at least two female characters talk to each other?

When a screenplay passes T1, then we know that it contains at least two named female characters. These named female characters could appear at any point within the screenplay, as T1 is concerned with only the number of female characters and not with how often they appear or if they appear together. Failing T2 is the most common requirement a movie fails, as we will see in Section 5.1. This makes developing an accurate method for determining who converses with whom especially important.

When watching a movie this is fairly easy to determine. Based on the dialogue and camera work a viewer usually knows who is talking to whom, unless the director purposefully hides this information from the viewer. Using only the screenplay to determine who is conversing with whom is trickier. How do we transform those subtle signals that appear on-screen into an accurate way to determine conversation partners?

For two characters to talk to each other, they need to firstly appear in the same scene together. Using the "S" lines in the parsed screenplays, we can determine when a scene begins and when a scene ends. The idea is to find these scene markers, create a network of which characters appear in which scene and number the order in which the characters exchange dialogue. Additionally some screenplays utilize "N" lines to break-up a scene, to describe something happening that does not include dialogue. If a screenplay contains these mid-scene "N" lines, we also use them as sub-scene breaks.

Based on the assumed gender results of T1, we can identify the scenes where two female characters were possibly part of the same conversation. These scenes then require further investigation. Scenes where the criteria for T1 were not met can be ignored, as they could not possibly cause a script to pass T2.

As we know from watching movies, just because two characters appear in the same scene together does not mean that they talk to each other. To solve this problem we adopted a conservative approach to analysing the characters and the order of their lines of dialogue. Characters only count as conversing when their dialogue lines follow each other consecutively. Each time an "S" line appears, a new scene starts. Within the "S" boundaries we are interested in the "N" and "C" lines. As described above, an "N" line breaks up a scene. We count each character attributed to a "C" line that appears consecutively within these sub-scenes as conversing with each other.

Say we find this sequence within a scene:

{S, N, C1, D, C2, M, D, N, C3, D, C1, D}

Using the conservative approach C1 and C2 converse and C3 converses with C1. The "N" line breaks up the scene, so that we start counting conversation partners fresh after it. This means that although C2 and C3 are next to each other in the conversation order, we do not count these characters as having exchanged dialogue.

A more liberal approach measures who converses with each other by looking only at which characters appear in a scene together. If they are in the same scene, that means each character says something at some point and then we say they talked to each other. In our example using this approach would mean that C1, C2, C3 all conversed with each other. This method proved to be too generous and allowed for many films to pass T2, although the film actually fails. Therefore we adopted the conservative method in the final version.

Algorithm 3.2 shows in pseudocode the steps described here for the conservative approach when determining if a screenplay passes T2. First we reduce the screenplay to its parsing labels and break the screenplay down into scenes. Each scene is broken into sub-scenes and within these sub-scenes the character's gender is compared to the previous character's gender. If both are female, then a conversation between at least two female characters has occurred and the screenplay passes T2. If no such instance can be found within the screenplay, the screenplay fails T2.

In order to better understand the results, we created a diagram to represent the network

**Algorithm 3.2:** T2: Do at least two women talk to each other?

```
Data: parsed screenplay, gender results
Result: number of dialogue exchanges between female characters
listParsedElements = reduceScriptToRelevantParsedElements(script) ;
byScene = breakDownByScene(listParsedElements) ;
numPass = 0 ;
for scene in byScene do
    for sub-scene in scene do
        prevChar = null ;
        for line in scene do
            nowChar = line ;
            if prevChar.gender = 'F' and nowChar.gender = 'F' then
                numPass += 1 ;
            end
            prevChar = line ;
        end
    end
end
```

of dialogue lines. The network shows every character who appears in the screenplay each as an individual node. If a line connects two characters, that means they exchanged dialogue at some point in the screenplay.

Figure 3.5 shows an example network that has been extracted from the screenplay of "Juno" (2007). The nodes that are connected with each other mean these characters have a conversation at some point in the film, according to our conservative definition of a conversation. This visualization method allows us to see immediately that the film revolves, unsurprisingly, around the character "Juno". There are certain clusters of other characters, but almost everyone interacts with her at some point. For T3 the features that can be calculated from the social network play a very important role.

## 3.6 T3: Is there at least one conversation between them about something other than a man?

The goal of T3 is to determine whether there is at least one conversation involving more than one named female character that is about something other than a man. This test is the most open to interpretation of the three tests, as determining the topic of a conversation is not always clear-cut. Men do not have to be explicitly mentioned for the conversation to be about a man and the reverse is also true. Simply because a man is mentioned does not mean that the conversation is about a man. When a character

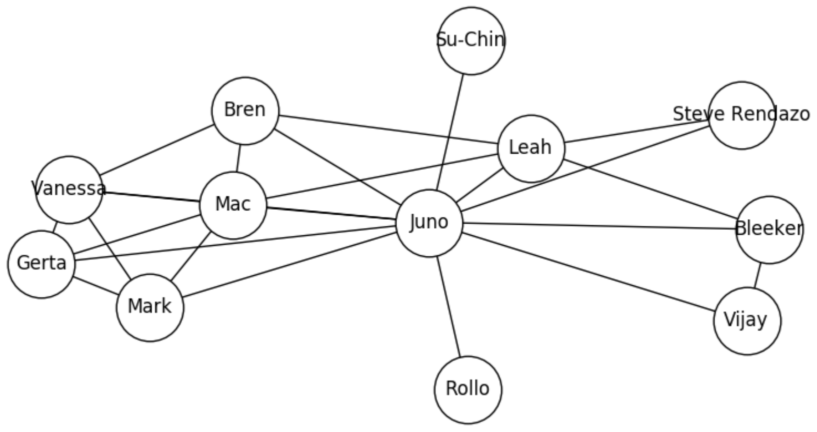


Figure 3.5: Social network of named characters from "Juno" (2007)

asks "What did Ryan say about your interview?", the topic of the conversation is the job interview that recently happened, not about the male character "Ryan". To automate T3 we developed and trained a machine learning model that uses linguistic features calculated from the dialogue text along with features found in the dialogue network of the screenplay.

In order to be able to estimate if a film passes T3, we calculated features using a variety of elements from the text. First the social network created to answer T2 in Section 3.5 provided the basis for many features. The "D" lines contain what is actually said, so they are naturally relevant when answering T3. Linguistic features like POS labeling are also used to determine what the dialogue is about.

At first we experimented with various topic models, like Latent Dirichlet Allocation, however we found topic models to be unfit for solving this task. A topic model clusters words found in the text into topics, however this does not tell us what the dialogue lines were about. The task has more similarities to a text classification problem. If we classify all dialogues to be either yes, about a man, or no, not about a man, we want to know if there is at least one "no" classification in dialogue between two female characters. We do not want to determine the topic of the conversation, but whether the conversation is about a man or not.

We selected the features based on the results obtained in Agarwal et al. [47]. It was shown that the features involving the analysis of the social network were the best indicators for T3. We added another category of features that pertained to general information about the specific film, like year of release and running time in minutes. Like in Agarwal et al. [47], the social network features performed very well.

### 3.6.1 Features

We defined four main types of features for our T3 model: BOW features (BOW), linguistic features (LING), social network analysis features (SNA) and film features (FILM) [47]. The focus is on interactions between female characters, since those are the most directly relevant for T3.

The BOW features are limited to the words that appear in the "D" lines and occur in conversations between pairs of female characters. Per film all words we attribute to "D" lines of these conversations are collected and assembled into a binary vector with a fixed length of 19,332. That way each word that appears only counts as one entry. This length was chosen, since that is the number of different words that occur between female characters in our training set. The number of conversations we can draw from varies for each movie, so using a binary vector means frequencies of word appearances are irrelevant. We only consider words that appears at some point in a conversation between female characters.

The LING features are concerned with general metadata about the conversations between the pairs of female characters. These are listed here:

- the number of conversations between the pairs of female characters
- the number of words exchanged during their conversations
- is there a mention of a male pronoun or male character in any of their conversations?
- is there a mention of a male pronoun or male character in all of their conversations?

As the number of pairs of conversing female characters varies from movie to movie, so does the length of the LING feature vector. If one movie has two pairs of conversing female characters and another has ten pairs of female characters, the feature length is longer for the second movie, since we analyse each pair of conversing female characters. In other words, the feature vectors length would be $\binom{n}{2}$. For this reason we converted the feature vectors to fixed length vectors by calculating the minimum, maximum, mean and standard deviation. After these normalization calculations we have 16 LING features, since for each of the four features we calculate these four values.

Next we looked at what SNA features we could define using the social network information we gained from T2. For each female character we calculate:

- degree centrality, or how many other characters are connected to the considered character
- closeness centrality, or sum of the length from the considered character to all other characters divided by the number of characters

- betweenness centrality, or a measure of how many of the shortest connections go "through" the considered character
- the number of men a female character is connected to
- how many other female characters are connected to this female character

As with the LING features, the length of the feature vectors can vary greatly from movie to movie, based on the number of characters. To get a feature vector that has a fixed length, we also normalized these features by calculating the minimum, maximum, mean and standard deviation, creating 20 SNA features.

We linked the film to the IMDb database for T1, so we experimented with using the information from their database to gain additional information about the film and created new FILM features. These features are the film's rating, year of release, genre and running time in minutes. To be used as features the values had to be numerical, therefore we converted the categorical features genre and rating using one-hot encoding. One-hot encoding has the advantage that no artificial numerical order is created, like with label encoding. For example, there are four possible ratings represented with the variables ratingG, ratingPG, ratingPG13 and ratingR. When a film has the rating 'G' it is represented by the variable ratingG having the value "1", with the other rating variables all having the value "0". We defined a catalogue of different genres and applied the same principle.

We then experimented with logistic regression models and support vector machines (SVM). We split the data into training and test sets. The size of the training and test sets is the same as in previous research [47], however the composition varies, since we did not have access to their training and test sets. In the end a combination of the features described above and a SVM model with a radial basis function (RBF) kernel performed the best, as we show in Section 5.2.

## 3.7 Final Approach Integration

We have described how to process each test and the final step is to combine them to get a final result. The composition is very straightforward. A screenplay document is first parsed and assessed on how well-structured it is. If the screenplay fails the parsing sanity checks, then it is not processed further. If it passes, then we move to T1.

The method applied for T1 is a combination of the classic dictionary approach and using external IMDb data to match a character name to the name used in IMDb. If a screenplay fails this requirement, then it gets a score of 0 out of 3. If it passes then it has a score of at least 1.

The screenplays that pass T1 then move onto T2. For T2 the interactions of the female characters are analysed closely and it is determined if the screenplay gets a passing grade or not. The screenplay is split up into its scenes and the order in which the characters speak is exported to a separate list. In the end we chose the conservative approach and the list is evaluated using this strategy. If no exchange in any scene fulfils the criteria, then the algorithm breaks here and the screenplay has a score of 1. If it passes, then it moves onto T3 and it gets a score of 2.

As the algorithm progresses, fewer and fewer sections of the screenplay remain relevant. After we know which characters are female, we are only interested in these characters. Scenes that involve only men or only one female character can be passed over.

For T3 the features calculated combine information from the social network, the actual text of the dialogue and general information about the specific film. The chosen model that performed the best was the SVM classifier with the RBF kernel. The SVM function is a binary classifier. If the result of the function is negative, then it is classified as '0', or failing T3. If the result is positive, then it is classified as passing T3 and therefore passing the Bechdel test with a score of 3.

CHAPTER 4

# Methodology for Novels

After extensively analysing screenplays and automatically testing whether they pass the Bechdel test, we pose the following question: is a similar analysis possible for novels? Novels contain more text than screenplays and have fewer formatting conventions. Can we obtain the same information from a novel as we did from a screenplay? As far as we can tell, no previous research exists in the area of using natural language processing techniques to determine whether a novel passes the Bechdel test.

For the Bechdel test the most fundamental elements are defining where a scene begins and ends, who talks within the scene, what gender do the characters have and what is spoken about within the dialogue. The idea of the Bechdel test is the same for screenplays and for novels, however the basic structures of the texts differ significantly. The two are so different, that decidedly different strategies have to be applied.

As we saw in Chapter 3, a screenplay is a well-structured document with many formatting elements built into it. All of those formatting rules become inapplicable as soon as we turn our sights to automating the Bechdel test for novels. For this reason, we must go about finding an algorithm that can be successful in a much different way.

## 4.1 Data

The classic resource for online available novels is Project Gutenberg [60]. This resource contains over 55,000 literary works from a variety of genres, languages and formats. We decided to focus on works from the nineteenth-century, as they are readily available online due to their expired copyright. Included are five works from authors Jane Austen, Charles Dickens and Sir Arthur Conan Doyle listed in the Table 4.1. The Bechdel score is sum of sub-tests passed (0-3). When no sub-test is passed then the Bechdel score is 0, if all criteria are fulfilled, then 3.

| Author | Title | Year | Nr. named speakers | Bechdel score |
|---|---|---|---|---|
| Jane Austen | *Pride and Prejudice* | 1813 | 18 | 3 |
| Jane Austen | *Emma* | 1815 | 26 | 3 |
| Charles Dickens | *A Christmas Carol* | 1843 | 11 | 3 |
| Sir Arthur Conan Doyle | *A Scandal in Bohemia* | 1888 | 11 | 1 |
| Sir Arthur Conan Doyle | *The Red-Headed League* | 1890 | 10 | 0 |

Table 4.1: Novels used in corpus

Since the difficulty of automating the Bechdel test is so much greater with novels, we have carefully chosen the novels to be analysed. Our goal was to find novels with a linear storytelling structure where the author uses chapters as scene boundaries as often as possible. No novel follows this rule all the time and dealing with these inconsistencies influenced how we developed our algorithm.

The ground truth was determined manually. As the number of texts being analysed is not so great, this was possible. However future larger scale analyses will most likely have to find a more time efficient way of generating the ground truth. As far as we were able to find, there is no available resource similar to “www.bechdeltest.com” for novels.

Three of the five novels that we examined pass the Bechdel test, although *A Christmas Carol* passes the third criteria based on one exchange quoted in Section 4.6 and the Jane Austen novels pass with no trouble. When testing only five works, written in the same time period, we cannot achieve a balanced sample. Due to the lack of annotated data, we leave the large-scale analysis for later research.

## 4.2 Format of a Novel

The formatting rules that helped us immensely when parsing screenplays do not exist for novels. However there are some basic formatting elements, like quotation marks, chapters and paragraphs, that we can use to find dialogue and scene boundaries.

A novel starts out with a title page, oftentimes a dedication, publishing information and maybe a table of contents. Sometimes the texts coming from Project Gutenberg also have additional lines of descriptive text about how long the text has been available on their website. These first few pages are irrelevant for us, but we need to be able to identify when the novel actually begins. A novel is then usually split up into chapters. We interpret the chapters to be the scenes in the novel.

The text within each chapter is broken down into paragraphs. The structure of the paragraphs varies greatly from novel to novel and even from chapter to chapter. There are paragraphs containing only descriptive text, paragraphs with a mix of dialogue sentences and description sentences and then paragraphs containing only dialogue. Sentences containing only description are most likely irrelevant for our task, however it is possible that there is some reference to a character is contained within a sentence and is relevant for a later piece of dialogue.

## 4.3 Parsing of Novel

When analysing a novel on the basis of the Bechdel test, we are interested in the same elements as with screenplays. We want to know when a scene begins and ends, who appears in the scene, who talks to whom and what do they talk about. Unlike with screenplays, for novels we do not assign each line or sentence a label in a parsing step.

A parsing algorithm similar to the one developed for screenplays is not practical, as there are so many possibilities and the information we need for each sub-test cannot be extracted from a single sentence or paragraph. Identifying quoted speech is fairly straightforward, because of the quotation marks surrounding the text. However to determine who said the dialogue and what their gender is, we must use more surrounding text than with screenplays. For example, if there is a line of dialogue without direct attribution we need the context surrounding the quote to determine who the speaker was.

Instead of a parsing algorithm that assigns labels for each line, we run NER, a POS tagger and coreference resolution over the entire text. The results are then incorporated into the algorithms for T1, T2 and T3, which is described in the respective sections below. We first apply NER to the entire text. We can limit the domain to only people, as the other possible labels do not interest us for this application. We used the Stanford implementation [61], interfaced by the Python package NLTK, since it is a commonly used NER in previous literature about character detection and quote attribution [16] [28].

As preparation we also ran a POS tagger over the entire text. For this task we choose the Stanford POS tagger [13]. Our focus is specifically on verbs, since we use the POS tags to help determine who said which lines of dialogue. The Stanford POS tagger [13] identifies, among other things, 6 different kinds of verbs, based on verb-tense. Since our application does not need to differentiate between tenses, we can ignore this detail and treat all tokens identified as verbs the same.

Below is an example of the output from the POS tagger, focusing on the verbs found in the text. The verb outside the quotation marks is highlighted in green and the verbs inside the quotation marks are highlighted in grey. The tagging label is found after the '/' symbol. There are approximately 45 different POS tags and to give an idea of the variety of possible tags, all tags in the example excerpt below have been included.

Original text:

> "I am afraid, Mr. Darcy," observed Miss Bingley in a half whisper, "that this adventure has rather affected your admiration of her fine eyes."

After POS tagging[1]:

> "/"I/PRP am/VBP afraid/JJ ,/, Mr./NNP Darcy/NNP ,/, "/" observed/VBD Miss/NNP Bingley/NNP in/IN a/DT half/JJ whisper/NN ,/, "/" that/IN this/DT adventure/NN has/VBZ rather/RB affected/VBN your/PRP$ admiration/NN of/IN her/PRP$ fine/JJ eyes/NNS ./. "/"

Another preparation step we took was to run coreference resolution over the entire text. For this we used another Stanford NLP library [62], with the deterministic system. The deterministic system is described in [62] and "is a collection of deterministic coreference resolution models that incorporate lexical, syntactic, semantic, and discourse information." The syntax of the result from the coreference resolution is more complicated to understand than the POS tagger. Say we want to run coreference resolution on this passage from *The Red-Headed League*:

> Holmes chuckled and wriggled in his chair, as was his habit when in high spirits. "It is a little off the beaten track, isn't it?" said he.

The result of the coreference resolution looks like this:

> ["Holmes" in sentence 1, "his" in sentence 1, "his" in sentence 1, "he" in sentence 3]

We parse the output to obtain arrays that are formatted like this:

> [[Holmes, 1, 1],[his,6,1],[his,11,1],[he,2,3]].

The first element in an array is the term that appears in the text, the second is the token number of appearance and the third is the sentence. The result links three mentions to "Holmes", that appear throughout the excerpt. We then connect all of these mentions, in case they are relevant for resolving T1 or T2, to "Holmes". During our tests of T1, we will see that "Holmes" is a character. At this stage however we are only interested in resolving the mentions that appear throughout the text.

How we use the results from NER, the POS tagger and coreference resolution is shown in the sections describing the methods developed for T1, T2 and T3. They provide the foundation for the algorithms described and are therefore vital to the developed method.

[1]PRP: Personal pronoun; VBP: Verb, non-3rd person singular present; JJ: Adjective; NNP: Proper noun, singular; VBD: Verb, past tense; IN: Preposition or subordinating conjunction; DT: Determiner; NN: Noun, singular or mass; VBZ: Verb, 3rd person singular present; RB: Adverb; VBN: Verb, past participle; NNS: Noun, plural

## 4.4 T1: Are there at least two named female characters?

The first step is to identify all of the named characters that appear at some point within the text. This is obviously more difficult than with screenplays, because in a novel there are no extra lines containing the name of the character who says the next lines of dialogue. It is also much more common in novels to refer to one character by different names. Using for example "Sherlock Holmes", "Mr. Holmes" and "Sherlock" within the same text is to be expected and our method needs to be able to link these different names to the same character. This is particularly difficult in novels like *Pride and Prejudice*, where so many characters have the same last name.

NER is the most common approach to detecting characters within a literary text. On its own however it was insufficient for our purposes. It has been shown that many characters go undetected when applying traditional NER systems [16], like the Stanford CoreNLP [15]. We cannot afford to have a low recall, as we need to identify all named characters that appear. We also need to connect names belonging to the same character, otherwise we run the risk of overestimating the number of different characters in the text.

Unlike with screenplays we are not able to utilize a database like IMDb that lists all characters that actually appear within the novel, since to our knowledge such a database does not exist. Resources like Wikipedia or Sparknotes tend to list only major characters or not provide a list of characters at all. For example the Wikipedia entry for Dicken's novel *A Christmas Carol* does not provide a list of characters that we could scrape. To create a ground truth we therefore manually compiled a complete list of characters for the novels that we analysed.

Recently researchers have been working on improving the traditional NER approach. We found the results from Vala et al. [16] to be the most promising in identifying characters and have applied the methods described for finding the named characters in the literary text. The goal of Vala et al. [16] was to identify every character, even minor characters who are never given a proper name. This goes further than is necessary for solving T1, so we removed the steps attempting to find these minor characters from the algorithm presented here.

We are only interested in the named characters and not the characters that appear in the text and are only referred to by nouns. For example in *The Red-Headed League* the character "girl" is referred to only by this noun and not a true name. If a character is only ever referred to as "a girl", then this character cannot cause a novel to pass T1. Pronouns also do not count as characters, since we want them to be referenced back to their "origin" names. While we are mining the text for character names and coreferences to resolve, we also look for clues about the gender of the characters. These clues appear in the form of pronouns and gender specific references.

The end result is a list of nodes, which are the names that appear in the text, and edges, which are connections between names that refer to the same character. This way when a name is used in the text, we can connect all of the aliases that actually reference the same character. Representing the result as a list of nodes and collection of edges is similar to SNA, which we applied when solving T2 for screenplays.

Below we describe step-for-step how the algorithm works that creates a list of character names from a literary text. The algorithm is first described in [16] with the only change made for this thesis is that the second to last step in the original description is not used, as that step adds characters who were not mentioned by name, but referred to by descriptors. These additional characters do not interest us, so we have eliminated that step for the algorithm. The idea of the algorithm is to start with a list of names found by running a NER over the text and then step-by-step connect names that most likely belong to the same character. Steps 1-4 add edges between nodes and steps 5-7 remove edges that we added too generously in previous steps.

1. Run a standard NER over the text. We used the Stanford NER implementation [15], described in Section 4.3. Each name found is handled as a separate name and each name can be thought of as a node. This sentence would lead to the creation of two nodes, "Mr. Bennet" and "Mr. Bingley":

   > [Mr. Bennet]$_{\text{Person}}$ was among the earliest of those who waited on [Mr. Bingley]$_{\text{Person}}$.

2. Run coreference resolution. This identifies the names that appear in a coreference chain and likely refer to the same character. For this we also used Stanford CoRef [62], the application of which we described in Section 4.3. We add edges between nodes found to be connected by coreference at any point in the text.

3. Add edges between nodes that, when removing an honorific, are exactly the same. For example the names "Mr. Holmes" and "Holmes" would be connected after this step, but also "Mr. Bennet" and "Mrs. Bennet".

4. Add edges between nodes of names that appear in the text, but are a variation of each other, like a nickname. An example would be "Elisabeth" and "Lizzy". We used a database found online [63] as the nickname dictionary.

5. We have now potentially added edges between names that are not truly aliases of the same character. Therefore we remove edges when one of these criteria is met:

   a) The genders after coreference resolution differ. For example if we have in the previous steps connected the nodes "Bennet" and "Elizabeth Bennet", but there is a coreference chain between "Bennet" and "he" and later a coreference "Elizabeth Bennet" and "she", with this step we would remove the edge connecting these nodes.

b) Names have same last name, but different first names. This is only applicable if the name contains more than one token, otherwise we only have a first or a last name.

c) An honorific between the two nodes differs. For example a name containing “Ms.” should not be connected to a name with “Mrs.”, as these likely refer to two separate characters.

6. Next we look at all pairs of nodes that are connected by an edge and remove any edges that meet one of these criteria:

   a) Both names appear together in a sentence, connected directly by a conjunction. For example if “Elizabeth and Jane” appears in the text, we would remove the edge between these two names.

   b) Both names appear together in quotation. If this occurs, then we can eliminate any edges between these characters.

7. Remove nodes that are disconnected from rest of graph and are also a portion of another name. These are usually ambiguous first or last names.

A small example is shown in Figure 4.1. The first graph shows the edges between the nodes after step 4. There are clearly edges connecting names that do not belong to the same character. The second graph shows the end result. Only the edge between the nodes “Lizzy” and “Elisabeth Bennet” remains.

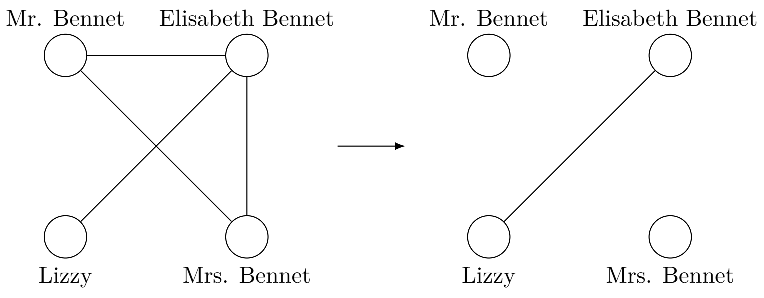


Figure 4.1: Example of algorithm to connect names that belong to same character. First graph shows the connections between nodes after step 4. The second graph shows the connections after all steps have been completed.

After we have a list of nodes and edges between names that appear in the text, we need to assign genders to those names. To determine the gender of the characters we first looked for hints in the name itself. Does one of the nodes contain a honorific like “Mr.” or are they referred to as someone’s aunt? Then we mined the coreferences in the vicinity of where the name appears in the text. For example in the text “Alex went to the swimming pool to meet up with his friends”, we know “Alex” is a male character, because he is

| Type | Example | Speaker |
|---|---|---|
| Explicit | “I do not cough for my own amusement,” replied **Kitty** fretfully. | Kitty Bennet |
| Pronominal | “No, I thank you,” **she** replied ... | Elizabeth Bennet |
| Nominal | “I should like it beyond anything!” said **her mother**. | Mrs. Bennet |
| Conversation | “What is his name?”<br>“Bingley.”<br>“Is he married or single?” | Mr. Bennet<br>Mrs. Bennet<br>Mr. Bennet |

Table 4.2: Examples of quotations

later referred to using a male pronoun.

The gender hints that are found within the text are extremely reliable. We also mined this information in the algorithm described above, as gender is an indicator if two character names belong to the same character or not. If we are not able to find any such hints, we use the dictionary approach, analogue as described in Section 3.4. For the characters central to the plot this additional step is hardly ever necessary, however if a character only appears briefly we have no other choice than to use external information for gender information.

The result is the same list of nodes with the various names that appear in the text, however this time with the added attribute of gender. All character name nodes connected together by edges of course have the same gender. Some major characters are connected to several names. For example in *Pride and Prejudice* the main character Elizabeth Bennet has nine different names she is referred to by over the course of the novel.

After we have this list of nodes and the corresponding gender it is simple to determine if a novel passes T1. Are there at least two female entries? If yes, then the novel passes T1; otherwise it fails.

## 4.5 T2: Do at least two female characters talk to each other?

Now that we have a list of candidate characters, including aliases, and their genders, we use the instances of dialogue and surrounding text to determine who speaks to whom at what point in the text. To provide an overview of the types of quotes that appear in literary texts, Table 4.2 shows some examples of the most common types of quotations and who the speaker of each example was.

Identifying the actual quote is easy enough; it is text that is surrounded by quotation marks. The problematic part is identifying the character, perhaps mentioned in the surrounding text, who said that particular quote. For this we look for clues in the text. One clue is finding the speech-verb [31]. A speech-verb, also known as reporting verb, is a verb that describes the speech. Classic examples are verbs like "to say", "to ask" and "to reply". Previous research about quote attribution [28] [30] list the most commonly found categories and combinations of quote, speech-verb and speaker for their specific datasets. The patterns used in this thesis are shown in the Table 4.3. Elson et al. [28] describes seven syntactic categories and introduces the patterns "Character trigram", "Anaphora trigram", "Dialogue chain", and "Conversation". The other three categories are used to classify quotations, but do not predict a speaker for the quotation. For this task we are only interested in patterns that assign a speaker. The more recent paper [30] added the additional patterns "Single mention" and "Paragraph final", along with the patterns from [28]. We ignore adverbs, adjectives and other irrelevant tokens when applying the patterns to find the speaker of each quote.

From the POS tagger that has been run over the entire text, described in Section 4.3, we know which tokens are verbs. We use this information to find the verbs in the surrounding text of the quote. As it is very possible that there are several verbs in the surrounding text, first a list of candidate speech-verbs for each chapter is created. A candidate speech-verb is any verb that appears in the text outside of a quotation. The possible verbs are scored, based on their proximity to the quoted text in question. The verb in the list of candidate speech-verbs that is closest is determined to be the speech-verb. Closeness is measured by distance in number of tokens and ignoring punctuation. In our sample set of five novels, most cases are fairly clear-cut. For example the speech-verb of the the following quote is "said", as it is the verb with the fewest tokens between it and the quote, namely 0.

> "You may depend upon it, Madam," **said** Miss Bingley, with cold civility, "that Miss Bennet will receive every possible attention while she remains with us."

In the case of a tie, then the verb that appears after the quote is declared the speech-verb.

All of the patterns used for quote attribution in this thesis have been defined in previous research, as described above. We know when a certain pattern appears in the text, which part of the pattern belongs to the "speaker" role and can then attribute the quote to this character. Table 4.3 lists the patterns and their definition. The notation of the definition is purposefully written as general as possible, since it describes the elements at the lowest possible level.

The goal of each pattern is to attribute a "<TARGET_QUOTE>" to a character or "speaker". CHAR_n is the character reference. Sometimes in the text, instead of an actual character name, only a “<PRONOUN>” is used, which is part of a coreference chain. “<SPEECH_VERB>” refers to the speech-verb, which we have identified as described above. “<OTHER_QUOTE>” refers to another quote in the text passage. Not the target quote, but a separate quote from the one we are currently attempting to attribute to a character.

| Pattern | Definition |
|---|---|
| Character trigram | <TARGET_QUOTE><SPEECH_VERB><CHAR_1><br>or <TARGET_QUOTE><CHAR_1><SPEECH_VERB><br>or <CHAR_1><SPEECH_VERB><TARGET_QUOTE> |
| Anaphora trigram | <TARGET_QUOTE><PRONOUN><SPEECH_VERB><br>or <TARGET_QUOTE><SPEECH_VERB><PRONOUN><br>or <PRONOUN><SPEECH_VERB><TARGET_QUOTE> |
| Dialogue chain | <OTHER_QUOTE by CHAR_1><TARGET_QUOTE> |
| Single mention | Within paragraph only one character is mentioned |
| Paragraph final | TARGET_QUOTE appears as last part of paragraph.<br>Quote attributed to final mention in same paragraph |
| Conversation | <OTHER_QUOTE by CHAR_1><br><OTHER_QUOTE by CHAR_2><br><TARGET_QUOTE by CHAR_1> |

Table 4.3: Quotation patterns

**Character trigram**: This is the most intuitive case of quote attribution. When a sentence like:

> “What have you done?” asked **Emma**.

appears in the text then it fits this pattern, or a variation. There is a quote, or text surrounded by quotation marks, followed by a speech-verb and then a character name. As language is flexible, the elements can be ordered in a variety of ways and still fit this pattern. CHAR_1 is the speaker of the target quote.

**Anaphora trigram**: When looking for anaphora trigrams we are looking for three elements appearing together: the quote we want to attribute, a pronoun and a speech-verb. The order of these three elements is flexible, like with the “Character trigram” pattern. When they appear together we attribute the quote to the character associated with the pronoun. For example:

> ... **she** said “I did not know before that you ever walked this way.”

**Dialogue Chain**: When we find this type of quotation in the text it means that we have identified the speaker of one quote and directly after that quote there is another quote. The line

> "No more have I," said **Mr. Bennet**; "and I am glad ..."

is an example of a dialogue chain. The first quote is identified at being spoken by "Mr. Bennet" and therefore the quote directly after is also spoken by him. CHAR_1 is the speaker in this pattern.

**Single mention**: A single mention is when looking at the paragraph that contains the quote, only one character is mentioned outside of the quotation. Here is an example of this from *Emma*, that fits this pattern and identifies "Mrs. Weston" as the speaker:

> **Mrs. Weston** looked surprised, and said, "I did not know that he ever had any such plan."

**Paragraph final**: If the quote appears at the end of the paragraph, then connect it to the last character mentioned in that paragraph. Here is an example of a quote that would be attributed to "Sherlock Holmes":

> ... **Sherlock Holmes**'s quick eye took in my occupation, and he shook his head with a smile as he noticed my questioning glances. "Beyond the obvious facts that he has at some time done manual labor, that he takes snuff, that he is a Freemason, that he has been in China, and that he has done a considerable amount of writing lately, I can deduce nothing else."

**Conversation**: Here we build on knowledge that we have already gained from the previous patterns. If a quote appears without a mention and surrounded by other quotes, we look at the conversation chain and find a quote that fits a previous pattern. This is our anchor for the surrounding conversation. Here is an example of a conversation:

> "His being so sure of succeeding was wrong," said **she**, "and certainly ought not to have appeared; but consider how much it must increase his disappointment!"
>
> "Indeed," replied **Elizabeth**, "I am heartily sorry for him; but he has other feelings, which will probably soon drive away his regard for me. You do not blame me, however, for refusing him?"
>
> "Blame you! Oh, no."

There are five quotes in this example. The first two quotes are spoken by "she", which we know from coreference resolution to be the character "Jane". Here the previously described rules Anaphora trigram and Dialogue Chain have already helped us correctly attribute this quote. These quotes would be "<OTHER_QUOTE by CHAR_1>" from Table 4.3. The next two quotes belong to "Elizabeth", which we know from the application of the Character Trigram and Dialogue Chain patterns. These are represented by "<OTHER_QUOTE by CHAR_2>" in the Conversation pattern. The next quote is therefore our "<TARGET_QUOTE by CHAR_1>". We attribute this quote to "Jane", since when applying this pattern we have identified her as CHAR_1.

If a quote fits one of these patterns, as about 68% of the quotes in our dataset of five novels do, then we attribute the quote according to the definition. If the quote does not lend itself to being attributed using a pattern, then we search the text for the character most frequently mentioned, either by name or per coreference, within a certain window of text. In other words we look at the text before and after the quote. The window used here was first used in Muzny et al. [30]. We counted the number of times a character is mentioned within the previous 2,000 tokens before the quote or 500 tokens after the quote. The character who is referred to most often within this window of tokens is then defined as the speaker of the quote.

After all quotes have been attributed to a certain character, we can build a social network analogue to what we described in Section 3.5. First we order the characters who we found to have said each quote chronologically. Scene boundaries are the chapters and we look for the characters who have dialogue lines following each other. This information is then used to determine if a novel passes T2. Do two female characters have consecutive quotes within a chapter? If we are able to find such a pattern, then the novel passes T2, otherwise it fails.

## 4.6 T3: Is there at least one conversation between them about something other than a man?

After defining who converses with whom, we need to determine if the conversations between female characters were only about men or involved other topics. From the algorithm for T2 we know who converses with whom during the course of the novel and now we need to further examine the conversations between the female characters.

When developing a method for solving T3 for screenplays, we divided our dataset into a training and a test set and applied a machine learning approach. Our sample of five novels is too small to sensibly apply a similar method for novels, especially considering T3 is only relevant for three of them. More annotated data is needed before a machine learning method can be applied to novels.

Instead of a machine learning approach, we used a rule-based approach. As with screenplays, we do not attempt to find the actual topic of the dialogue, but how likely is it that at least one conversation exists that is not about a man. We defined our own rules to determine if a novel contains such an exchange.

For each quote we determined whether the quote contains a reference to a male character or a male pronoun like “he”, by using the data we acquired for solving T1. This is not an ideal measurement, since conversations can take place that are about a man, but do not reference them directly. Yet in our sample set these two features proved to be a good indicator for T3. We tested many different features, including using the social network from T2 and general features about the number of conversations between the female characters. In the end, these two features were the best predictors.

*A Christmas Carol* passes T3 based on this exchange between "Mrs. Cratchit" and her daughter "Martha":

> "Why, bless your heart alive, my dear, how late you are!" said Mrs Cratchit, kissing her a dozen times, and taking off her shawl and bonnet for her with officious zeal.
>
> "We'd a deal of work to finish up last night," replied Martha, "and had to clear away this morning, mother."

The method described here correctly identifies this exchange as not being about a man, since no male characters are referenced. In the same chapter there is another exchange between these characters that does not pass T3:

> "What has ever got your precious father then?" said Mrs Cratchit. "And your brother, Tiny Tim; And Martha warn't as late last Christmas Day by half-an-hour."
>
> "Here's Martha, mother," said Martha, appearing as she spoke.

This exchange does not pass our T3 test, since male characters are mentioned. It also underscores how tricky it is even as a reader to determine the topic of a sentence or quote. Is this more about Christmas, being late or about who is late? For our sample set this approach worked well in that it correctly identified three novels as passing T3.

## 4.7 Final Approach Integration

The structure of the final approach is very intuitive. First the text of the novel is run through the parsing steps. Here no text can fail and therefore all texts are then run through the T1 algorithm. After identifying all named characters and their genders, we count the number of female characters. If there are more than two, then the novel moves on to T2. Otherwise the text receives a Bechdel score of 0 and fails the Bechdel test.

The algorithm for T2 requires us to first attribute each quote to a speaker. After this has been done, we build the order of the conversations that take place in each chapter. If two female characters have consecutive dialogue lines within a chapter, then the text passes T2 and moves on to T3. If no such pattern can be found, then the text fails T2 and receives a Bechdel score of 1.

After the novel passes T2 we move on to T3. We use the conversations identified in T2 as being between two female characters to calculate the T3 features. If for each conversation one the features is true, as in a reference to a male character or pronoun exists in the dialogue, then the novel fails T3. If both features are false for any conversation between female characters, the novel passes T3 and therefore the Bechdel test.

CHAPTER 5

# Results

Now that we have described the methodologies used to automatically calculate the Bechdel test result for screenplays and novels, we need to determine how well our methods perform. For screenplays we are able to use the crowd-sourced website “www.bechdeltest.com” as our ground truth. Every film we analysed has an entry, with a score for T1, T2 and T3. There is no similar website for novels, so we manually determined the Bechdel score of all the novels in our corpus.

## 5.1 General Bechdel Results for Films

Before actually implementing an automatic method for solving the Bechdel test, we analysed the large amount of data already available about which films pass the Bechdel test and which fail in order to get a better understanding about what factors could be most important. The basis for this analysis were the crowd-sourced results found on the website “www.bechdeltest.com”. This website contains a record of films and their Bechdel scores, including a Boolean uncertainty parameter “dubious”. Using their API and matching the results with records from IMDb, we were able to create a database of approximately 1,800 films. The selected films were all made after 1970 and earned over 2 million dollars, to focus on movies that likely had a wide release. We also eliminated all films with a Bechdel score that had been labeled by a contributor as “dubious”.

When interpreting the results, shown in Figure 5.1, the most important finding was that the second requirement that two named women talk to each other at any point in the film was the likeliest reason for failing the Bechdel test. In Figure 5.1 we see that the overall percentage of films that fail the Bechdel test has decreased since the 1970’s, although the percentage of films that fail has not changed dramatically within the last few decades. Failure to pass T2 is consistently the most common reason a film fails the Bechdel test. From all the films we looked at, on average 24% of the time a film is unable

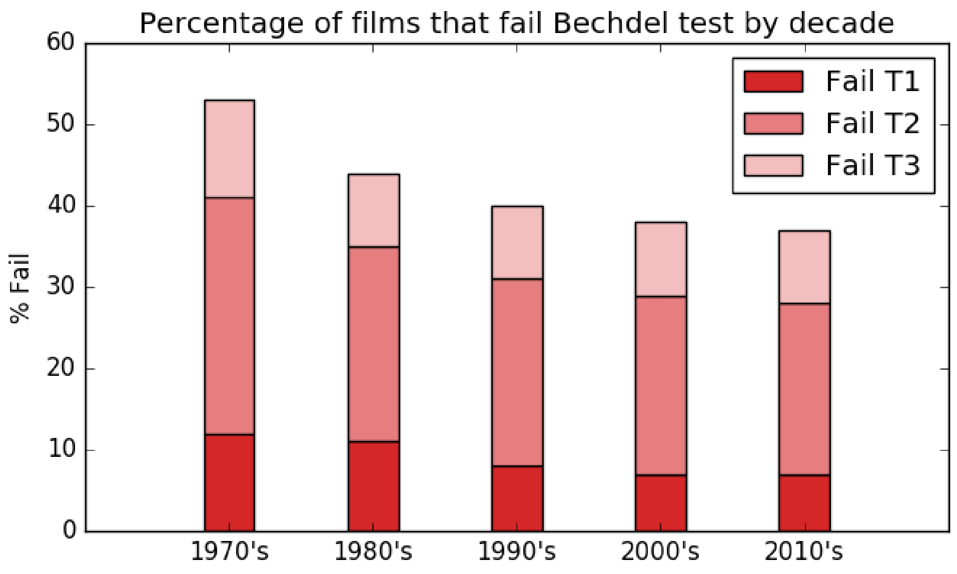


Figure 5.1: Percentage of films that fail the Bechdel test by decade

to pass T2. The failure rates for T1 and T3 are similar and on average about 9% and 10%, respectively.

## 5.2 Results for Screenplays

The composition of the corpus that we obtained from the authors of [47] is shown in Table 5.1. Each criterion of the Bechdel test is shown individually and split by whether the screenplay passes or fails the criterion. The database contains 457 screenplays in total, where 266 fail the Bechdel test and 191 pass.

| | Fail | Pass |
|---|---|---|
| Bechdel T1 | 31 | 426 |
| Bechdel T2 | 160 | 266 |
| Bechdel T3 | 75 | 191 |
| Total | 266 | 191 |

Table 5.1: Composition of screenplay corpus

When evaluating the described methods, we split the dataset into the screenplays that fail T1 and those that pass T1. For each class we calculated the P (precision), R (recall) and F1 scores. We also show here a confusion matrix for each method applied. For example the “Dict” method results in 27 films being classified as failing T1. From these 27 films, only 8 actually failed T1, with a ground truth of 31 films in our dataset failing T1. The “Dict” method therefore results in 430 screenplays being identified as passing T1, where 407 actually pass T1, with a ground truth of 426 films passing T1. This is shown in Table 5.2. The remaining confusion matrices are shown in Table 5.3 and Table 5.4. Since we calculated two F1 scores for each method, one for the films that pass T1 and one for the films that fail, we additionally calculated a Marco-F1, to give an overall evaluation of each method. The Macro-F1 weighs each class equally.

| | | Prediction | | |
|---|---|---|---|---|
| | | Pass | Fail | Total |
| | Pass | 407 | 19 | 426 |
| Actual | Fail | 23 | 8 | 31 |
| | Total | 430 | 27 | 457 |

Table 5.2: Confusion matrix for “Dict” method

| | | Prediction | | |
|---|---|---|---|---|
| | | Pass | Fail | Total |
| | Pass | 410 | 16 | 426 |
| Actual | Fail | 18 | 13 | 31 |
| | Total | 428 | 29 | 457 |

Table 5.3: Confusion matrix for “IMDb” method

| | | Prediction | | |
|---|---|---|---|---|
| | | Pass | Fail | Total |
| | Pass | 420 | 6 | 426 |
| Actual | Fail | 17 | 14 | 31 |
| | Total | 437 | 20 | 457 |

Table 5.4: Confusion matrix for “Dict + IMDb” method

For screenplays we implemented two methods of assigning a gender to a name: the dictionary approach (“Dict”) and the IMDb approach. We found that these methods worked best in combination, using the algorithm described in Section 3.4, and this is reflected in the results shown in Table 5.5.

| | Fail T1 | | | Pass T1 | | | |
|---|---|---|---|---|---|---|---|
| Method | P | R | F1 | P | R | F1 | Macro-F1 |
| Dict | 0.30 | 0.26 | 0.28 | 0.95 | 0.96 | 0.95 | 0.61 |
| IMDb | 0.45 | 0.42 | 0.43 | 0.96 | 0.96 | 0.96 | 0.70 |
| Dict + IMDb | 0.70 | 0.45 | 0.55 | 0.96 | 0.99 | 0.97 | 0.76 |
| Agarwal et al. [47] | 0.52 | 0.55 | 0.54 | 0.97 | 0.96 | 0.96 | 0.75 |

Table 5.5: T1 results for screenplays, by method

Combining the IMDb and Dictionary approaches provided the best results. This is likely due to the fact the the two methods have very different focal points. The dictionary approach looks at a name independently of the movie it is found in while the IMDb method focuses exclusively on the specific movie and its attributed credits. This combination allows for unambiguous cases to be handled quickly, while also giving a back-up strategy in case the gender of the name is uncertain.

The results for T1 in Table 5.5 are also consistent with the results from Agarwal et al. [47]. The last row in Table 5.5 shows the results from their research. We achieved similar F1 scores using only the character names found in the screenplay. This was possible because of the algorithm developed to combine the two approaches. The logic of when to use which approach and when the results are too uncertain to assign a gender is more sophisticated than what was previously used.

For T2 we tested two methods of finding conversation partners. First the conservative method where characters only count as conversing when their dialogue lines follow one another directly. The second method was more liberal, where the characters only had to have a line of dialogue in the same scene as another character to count as having conversed. Our results are shown in Table 5.8, where we again separated the screenplays into one class that fail T2 and another that pass T2. The confusion matrices in Table 5.6 and Table 5.7 are also shown here. A Macro-F1 score is then calculated, to give an overall evaluation. It is clear that the conservative approach leads to results with a higher F1 score.

| | | Prediction | | |
|---|---|---|---|---|
| | | Pass | Fail | Total |
| Actual | Pass | 210 | 56 | 266 |
| | Fail | 124 | 36 | 160 |
| | Total | 334 | 92 | 426 |

Table 5.6: Confusion matrix for “Liberal” method

| | | Prediction | | |
|---|---|---|---|---|
| | | Pass | Fail | Total |
| Actual | Pass | 237 | 29 | 266 |
| | Fail | 112 | 48 | 160 |
| | Total | 349 | 77 | 426 |

Table 5.7: Confusion matrix for "Conservative" method

| | Fail T2 | | | Pass T2 | | | |
|---|---|---|---|---|---|---|---|
| Method | P | R | F1 | P | R | F1 | Macro-F1 |
| Liberal | 0.39 | 0.23 | 0.29 | 0.63 | 0.79 | 0.70 | 0.49 |
| Conservative | 0.62 | 0.30 | 0.41 | 0.68 | 0.89 | 0.77 | 0.59 |
| Agarwal et al. [47] | 0.63 | 0.28 | 0.39 | 0.67 | 0.90 | 0.77 | 0.58 |

Table 5.8: T2 results for screenplays, by method

The results for T2, shown in Table 5.8 are very similar to the results achieved by Agarwal et al. [47]. This was to be expected, however, since we did not noticeably vary from the approach described in their paper. The conservative approach was there also much better at correctly answering T2.

In our testing of T3 we need to define a training set and an unseen test set. We used the same proportions for training and testing as [47]. Our training and validation set consisted of 60 films that fail the Bechdel test and 153 that pass. This leaves 15 films that fail and 38 that pass for our test set. Table 5.9 shows the averaged 5-fold cross-validation P/R/F1 measures for the best combination of features and classifiers. Since our training and validation set is unbalanced, meaning there is not an equal amount of cases for failing and passing the Bechdel test, we penalized mistakes in the minority class more than those in the majority class. Mistakes made in the minority class were penalized by a factor of 2.55 (153/60). Mistakes in the majority class were penalized by a factor of 1. Having an unbalanced training set with a 0-1 loss function can lead to a classifier learning a function that is optimizing for accuracy and has a bias to the majority class, which we attempted to avoid by using this penalization system.

| | | Fail T3 | | | Pass T3 | | | |
|---|---|---|---|---|---|---|---|---|
| Method | Kernel | P | R | F1 | P | R | F1 | Macro-F1 |
| SVM | Linear | 0.38 | 0.63 | 0.47 | 0.80 | 0.59 | 0.68 | 0.57 |
| SVM | RBF | 0.44 | 0.68 | 0.53 | 0.84 | 0.66 | 0.74 | 0.64 |
| Agarwal et al. [47] | | 0.42 | 0.84 | 0.56 | 0.90 | 0.55 | 0.68 | 0.62 |

Table 5.9: T3 results for screenplay, by method

In Table 5.9 we show the results of the SVM method using two different kernels and for comparison the results from Agarwal et al. [47]. The method SVM with the RBF kernel achieves a higher Macro-F1 score and performs slightly better in this regard than the previous method. The T3 features with the highest correlation were the SNA features, more than the features that included information about the actual dialogue. The social network and the measurements of centrality appear to be the best indicators for this task. This would indicate that movies that fail the Bechdel test are more likely to have female characters who are connected less to the other characters.

Now that we have looked at each component of the Bechdel test individually, we can evaluate the task as a whole. We created a test set of 38 movies that passed and 52 movies that failed the Bechdel test. Here we are only interested in whether the final method is able to correctly classify a screenplay as passing the Bechdel test or not. If a movie fails T1, therefore failing the Bechdel test, but we incorrectly say the movie passes T1 and T2, only to fail T3, that does not make a difference for the end task. When evaluating the end task, it is only important that the film has been correctly classified as failing the Bechdel test. That the sub-tests were partially incorrect does not matter in this case.

| | | Prediction | | |
|---|---|---|---|---|
| | | Pass | Fail | Total |
| Actual | Pass | 32 | 6 | 38 |
| | Fail | 11 | 41 | 52 |
| | Total | 43 | 47 | 90 |

Table 5.10: Confusion matrix for test set for T1 + T2 + T3

| | Fail Bechdel Test | | | Pass Bechdel Test | | | |
|---|---|---|---|---|---|---|---|
| Method | P | R | F1 | P | R | F1 | Macro-F1 |
| T1+T2+T3 | 0.87 | 0.79 | 0.83 | 0.74 | 0.84 | 0.79 | 0.81 |
| Agarwal et al. [47] | 0.80 | 0.91 | 0.85 | 0.83 | 0.66 | 0.73 | 0.79 |

Table 5.11: Results for overall screenplay approach

Our final method used Dict + IMDb for T1, Conservative for T2 and SVM with the RBF kernel for T3. Table 5.11 shows the end result of the method described in this thesis and for comparison the best results from the Agarwal et al. [47] paper. The Macro-F1 score is slightly higher and our method had much higher recall in the instances where the movie passes the Bechdel test.

## 5.3 Results for Novels

When looking at the results of the methodology defined to automate the Bechdel test for novels, we are dependent on manual annotations prepared specifically for this thesis and sometimes annotations done by other researchers and were made publicly available by them. We need a dataset that is usable for testing all three sub-tests and this limited the texts we could include. The five texts we used, listed in Table 5.12, are texts that have been used in previous research relevant for the sub-tasks. The Jane Austen novels *Pride and Prejudice* and *Emma* were manually annotated for [30] and *Pride and Prejudice* and the Sherlock works were used in [16] for determining the characters that appear in novels. The novel *A Christmas Carol* was used in [28], along with *Emma* and the Sherlock works.

The approach we defined for automating T1 involves finding all the names that appear within the text, then connecting the names that belong to the same character and using the surrounding text to determine the gender of the character. We manually compiled a list of all the named characters in each novel, as well as the character's gender. Then we ran our algorithm over the entire text of the novel and compared it's results with our manually compiled list.

| Novel | P | R | F1 | Pass T1 | Predicted correctly | Nr. named characters |
|---|---|---|---|---|---|---|
| *Pride and Prejudice* | 0.64 | 0.82 | 0.72 | Y | Y | 54 |
| *Emma* | 0.73 | 0.69 | 0.71 | Y | Y | 78 |
| *A Christmas Carol* | 0.59 | 0.68 | 0.63 | Y | Y | 14 |
| *A Scandal in Bohemia* | 0.47 | 0.64 | 0.54 | Y | Y | 11 |
| *The Red-Headed League* | 0.50 | 0.60 | 0.55 | N | Y | 13 |

Table 5.12: T1 results for novels

Table 5.12 shows the results for each novel. Of the texts in our sample, only *The Red-Headed League* fails T1 and this is correctly identified by the algorithm. The only named female character is "Miss Mary Sutherland", who is mentioned several times, but is the only named female character. All of the other texts pass T1 and this is also correctly identified in each case. Since the female characters often appear with an honorific, making it easy to identify the character's gender, this was not surprising. Table 5.12 shows how many more characters appear in the Austen novels than the other texts. There are so many female characters in her novels that they both pass T1 within the first chapter.

Many of the errors in finding characters had their origin in the NER, classifying minor characters like "Mrs. Long" simply as proper nouns. The novel *Pride and Prejudice* presented numerous challenges for correctly connecting names used in the text for the same character. Several characters share last names and the character "Catherine Bennet" even shares a first name with another character "Lady Catherine". "Catherine Bennet" is sometimes referred to by her nickname "Kitty", which was not a part of the nickname database used [63], making this connection go undetected by the algorithm. Another interesting problem was posed by the novel *A Christmas Carol.* Three characters are called "Ghost of Christmas Past", "Ghost of Christmas Present" and "Ghost of Christmas Yet to Come". The algorithm was not able to find these character names and this caused problems with the later steps, since it was then difficult to attribute dialogue to these characters.

For T2 we needed to determine which characters talked to each other throughout the novel. We must first establish which characters said which lines of dialogue. After we know who said what, we can apply a similar approach as with the screenplays. We only have annotated data at a quote level for *Pride and Prejudice* and *Emma.* Therefore we only calculate P/R/F1 scores for these texts. The accuracy for predicting T2 can be calculated for all texts that passed T1 and is shown in Table 5.13.

| Novel | P | R | F1 | Pass T2 | Predicted correctly |
|---|---|---|---|---|---|
| *Pride and Prejudice* | 0.74 | 0.70 | 0.72 | Y | Y |
| *Emma* | 0.68 | 0.74 | 0.71 | Y | Y |
| *A Christmas Carol* | - | - | - | Y | Y |
| *A Scandal in Bohemia* | - | - | - | N | Y |

Table 5.13: T2 results for novels

The most interesting text for T2 was the Dicken's novel *A Christmas Carol*, since there is only one scene where female characters exchange dialogue. As the quotes in this scene were all classic examples of "Character trigram" and "Dialogue chain", the speakers were correctly identified by the algorithm, therefore correctly predicting that the novel passes T2.

Three of the novels in our sample pass T2 and all of these also pass T3. Table 5.14 shows these results. We were able to develop a method that worked very well for our small sample set of texts. For this task, features that use the actual dialogue worked very well.

| Novel | Pass T3 | Predicted correctly |
|---|---|---|
| *Pride and Prejudice* | Y | Y |
| *Emma* | Y | Y |
| *A Christmas Carol* | Y | Y |

Table 5.14: Accuracy of T3 method for novels

In the end we developed a method that achieved complete accuracy for automating the Bechdel test for five novels. The methods developed to solve T1 and T2 were very sophisticated and since the Austen novels so clearly pass the Bechdel test and the Sherlock works obviously don't, less sophisticated methods would perhaps have been sufficient.

We also ran the described algorithm on three novels written in the 20th century. For each work we manually determined if it passes the Bechdel test and then compared this with the algorithm's result. *The Great Gatsby* (1925) passes the Bechdel test, but only due to a few passages. Since these followed classic patterns, the algorithm was able to identify them. The novel *The Emerald City of Oz* (1910) also passes the Bechdel test. Here the main character Dorothy interacts with several other female characters, like Aunt Em and Billina. Billina is an animal companion, a chicken, although it is referred to throughout the text as if it were a person and is able to converse with the other characters. This novel does not make a lot of use of honorifics, therefore gender determination of the characters relied heavily on the coreferences used throughout the text. The Western novel *The Virginian* (1902) does not pass the Bechdel test. The algorithm correctly identifies this, although it finds more female characters than there actually are in the novel, since the animals are sometimes referred to as "she" and with coreference resolution the "she" is resolved to the name of the animal. Each novel presents it's own challenges and although overall the algorithm often accurately determines if a novel passes or fails the Bechdel test, the details reveal the level of difficulty in the sub-tasks.

CHAPTER 6

# Conclusion

The impact of computational methods on the literary field will continue to spread as methods become more advanced and more annotated data is made available. The task of automating the Bechdel test for two vastly different types of text allowed us to explore many state-of-the-art NLP methods. It also showed us the limitations of those methods.

The Bechdel test was originally conceived as a test for movies, with three basic requirements. By applying NLP methods to screenplays we were able to gain insights into the structure of this type of document. We focused on the essential components needed to answer each criterion and were able to design strategies for solving each Bechdel requirement. For screenplays we devised a new rule-based approach for parsing screenplays and a new algorithm for assigning a gender to character names. Due to the previous literature that also attempted to automate the Bechdel test [47], we were able to compare our results for screenplays to their results.

After looking at screenplays we adopted what we learned to novels. There has been no previous research about using NLP to assess if a novel passes or fails the Bechdel test. In the process of transforming the knowledge gained by analysing screenplays to novels, we realized the two types of text required significantly different strategies. Without the well-structured basis of a screenplay, we needed to apply different strategies to novels. There has been some recent research in the area of finding characters and attributing quotes within literary texts that we were able to adapt and incorporate into our methods. The method we developed was able to correctly predict the Bechdel score of all five novels in our dataset.

The lack of information pertaining to novels that pass or fail the Bechdel test restricted our ability to test a wide variety of literary texts. Additionally the limited availability of annotated data limited out ability to apply machine learning techniques. However we used what was available, sometimes creating our own data, and were able to develop relatively successful strategies for all three requirements.

The amount of available data is constantly increasing, as is the quality of this data. The field of NLP will utilize the ever increasing amount of data to improve already existing techniques and develop new ones. For example where researchers used to painstakingly read through a novel to identify all the characters that appear and their aliases, now a program can help make that researcher more efficient.

Future research will most likely invest resources into more machine learning methods. This means annotating screenplays and novels to create appropriate training and test data. In future work we would focus on literary texts and completing a large-scale analysis of a wider variety of novels. Using a larger dataset, future researchers could evaluate if SNA features are similarly good T3 predictors for novels as for screenplays. Designing a machine learning strategy could lead to new breakthroughs in automating the Bechdel test. Also applying unsupervised learning methods would be the logical first step, if we are trying to bypass the issue of annotated data.

# Bibliography

[1] Alison Bechdel. The Rule. In Dykes to Watch Out For, 1985.

[2] Chris Taylor. Was ‘Rogue One’ good for female representation? Not so much, study says, January 2017. `www.mashable.com/2017/01/17/rogue-one-female-roles/`. Accessed 05.02.2017.

[3] Anu Koivunen, Ingrid Ryberg, and Laura Horak. Swedish cinema’s use of the Bechdel test is a provocation that works. The Guardian. `http://www.theguardian.com/commentisfree/2013/nov/27/swedish-cinema-bechdel-test-works`. Accessed 07.09.2014.

[4] Arit John. Beyond the Bechdel Test: Two (New) Ways of Looking at Movies. The Atlantic. `https://www.theatlantic.com/entertainment/archive/2013/08/beyond-bechdel-test-two-new-ways-looking-movies/311930/`. Accessed 31.12.2017.

[5] Nikesh Shukla. After the Bechdel Test, I propose the Shukla Test for race in film. NewStatesman. `https://www.newstatesman.com/2013/10/after-bechdel-test-i-propose-shukla-test-race-film`. Accessed 31.12.2017.

[6] David Garcia, Ingmar Weber, and Venkata Rama Kiran Garimella. Gender asymmetries in reality and fiction: The bechdel test of social media. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014.*, 2014.

[7] Nitin Indurkhya and Fred J. Damerau. *Handbook of Natural Language Processing*. Chapman & Hall/CRC, 2nd edition, 2010.

[8] A. M. Turing. Computing machinery and intelligence. *Mind*, 59(236):433–460, 1950.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[10] Åse Dragland. Big Data, for better or worse: 90% of world’s data generated over last two years. `www.sciencedaily.com/releases/2013/05/130522085217.htm`, May 2014. Accessed 01.10.2017.

[11] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ÅĄukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

[12] Michael Crawford, Taghi M. Khoshgoftaar, Joseph D. Prusa, Aaron N. Richter, and Hamzah Al Najada. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):23, Oct 2015.

[13] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[14] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26, January 2007. Publisher: John Benjamins Publishing Company.

[15] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[16] Hardik Vala, David Jurgens, Andrew Piper, and Derek Ruths. Mr. Bennet, his coachman, and the Archbishop walk into a bar but only one of them gets recognized: On The Difficulty of Detecting Characters in Literary Texts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 769–774, 2015.

[17] Mariona Coll Ardanuy and Caroline Sporleder. Structure-based clustering of novels. In *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*, pages 31–39, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

[18] Benjamin Mako Hill and Aaron Shaw. The wikipedia gender gap revisited: Characterizing survey response bias with propensity score estimation. *PLoS One*, 2013.

[19] Morgane Ciot, Morgan Sonderegger, and Derek Ruths. Gender inference of twitter users in non-english contexts. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1136–1145. Association for Computational Linguistics, 2013.

[20] Shlomo Argamon and Anat Rachel Shimoni. Automatically categorizing written texts by author gender. *Literary and Linguistic Computing*, 17:401–412, 2003.

[21] David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. Gender in Twitter: Styles, stances, and social networks. *CoRR*, abs/1210.4567, 2012.

[22] Martin Grandjean. A social network analysis of Twitter: Mapping the digital humanities community. *Cogent Arts & Humanities*, 3(1), 2016.

[23] Leon Danon, Ashley P Ford, Thomas House, Chris P Jewell, Matt J Keeling, Gareth O Roberts, Joshua V Ross, and Matthew C Vernon. Networks and the epidemiology of infectious disease. *Interdisciplinary perspectives on infectious diseases*, 2011, 2011.

[24] Linton C. Freeman. *The Development of Social Network Analysis: A Study in the Sociology of Science*. BookSurge Publishing, 2004.

[25] J.L. Moreno, H.H. Jennings, E.S. Whitin, and National Committee on Prisons. *Group method and group psychotherapy*. Sociometry monographs. Beacon House, 1932.

[26] Leslie Ball. Automating social network analysis: A power tool for counter-terrorism. *Security Journal*, 29(2):147–168, Apr 2016.

[27] Ralf Krestel, Sabine Bergler, and René Witte. Minding the Source: Automatic Tagging of Reported Speech in Newspaper Articles. In European Language Resources Association (ELRA), editor, *Proceedings of the Sixth International Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, Morocco, May 2008.

[28] David K. Elson and Kathleen McKeown. Automatic Attribution of Quoted Speech in Literary Narrative. In Maria Fox and David Poole, editors, *AAAI*. AAAI Press, 2010.

[29] David K. Elson, Nicholas Dames, and Kathleen McKeown. Extracting Social Networks from Literary Fiction. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *ACL*, pages 138–147. The Association for Computer Linguistics, 2010.

[30] Grace Muzny, Michael Fang, Angel X. Chang, and Dan Jurafsky. A two-stage sieve approach for quote attribution. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 460–470, 2017.

[31] Kevin Glass and Shaun Bangay. A naive salience-based method for speaker identification in fiction books. In *In PRASA 2007: Proceedings of the 18th Annual Symposium of the Pattern Recognition Association of South Africa*, pages 1–6, 2007.

[32] William S. Niederkorn. A Scholar Recants on His 'Shakespeare' Discovery. http://www.nytimes.com/2002/06/20/arts/a-scholar-recants-on-his-shakespeare-discovery.html, June 2002. Accessed 30.08.2017.

[33] Matthew Kirschenbaum and Martin Mueller. The remaking of reading: Data mining and the digital humanities. NGDM 07, National Science Foundation, 2007.

[34] Matthew L. Jockers. *Macroanalysis: Digital Methods and Literary History*. University of Illinois Press, 2013.

[35] Jacques Savoy. The federalist papers revisited: A collaborative attribution scheme. *Proceedings of the American Society for Information Science and Technology*, 50(1):1–8, 2013.

[36] Cristian Danescu-Niculescu-Mizil and Lillian Lee. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*, 2011.

[37] D. Trottier. *The Screenwriter's Bible: A Complete Guide to Writing, Formatting, and Selling Your Script*. Silman-James Press, 1998.

[38] C. Riley. *The Hollywood Standard: The Complete and Authoritative Guide to Script Format and Style*. Michael Wiese Productions, 2009.

[39] Eric Hoyt, Kevin Ponto, and Carrie Roy. Visualizing and Analyzing the Hollywood Screenplay with ScripThreads. *Digital Humanities Quarterly*, 8(4), 2014.

[40] Cristian Danescu-Niculescu-Mizil, Justin Cheng, Jon Kleinberg, and Lillian Lee. You had me at hello: How phrasing affects memorability. In *Proceedings of ACL*, pages 892–901, 2012.

[41] Annalee Newitz. Movie written by algorithm turns out to be hilarious and intense, June 2016. https://arstechnica.com/gaming/2016/06/an-ai-wrote-this-movie-and-its-strangely-moving/. Accessed 04.02.2017.

[42] Jehoshua Eliashberg, Sam K. Hui, and Z. John Zhang. From story line to box office: A new approach for green-lighting movie scripts. *Management Science*, 53(6):881–893, 2007.

[43] Stephan Heyman. Google Books: A Complex and Controversial Experiment. https://www.nytimes.com/2015/10/29/arts/international/google-books-a-complex-and-controversial-experiment.html, Oct 2015. Accessed 30.08.2017.

[44] Byron Wallace. Multiple Narrative Disentanglement: Unraveling Infinite Jest. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1–10, Montréal, Canada, June 2012. Association for Computational Linguistics.

[45] Andrew J. Reagan, Lewis Mitchell, Dilan Kiley, Christopher M. Danforth, and Peter Sheridan Dodds. The emotional arcs of stories are dominated by six basic shapes. *EPJ Data Science*, 5(1):31, Nov 2016.

[46] F. Moretti. *Atlas of the European Novel, 1800-1900.* Verso, 1999.

[47] Apoorv Agarwal, Jiehan Zheng, Shruti Vasanth Kamath, Sriram Balasubramanian, and Shirin Ann Dey. Key Female Characters in Film Have More to Talk About Besides Men: Automating the Bechdel Test. In *NAACL 2015*, 2015.

[48] Anil Ramakrishna, Nikolaos Malandrakis, Elizabeth Staruk, and Shrikanth S. Narayanan. A quantitative analysis of gender differences in movies using psycholinguistic normatives. In *EMNLP*, pages 1996–2001. The Association for Computational Linguistics, 2015.

[49] Tanaya Guha, Che-Wei Huang, Naveen Kumar, Yan Zhu, and Shrikanth S. Narayanan. Gender representation in cinematic content: A multimodal approach. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 31–34, New York, NY, USA, 2015. ACM.

[50] Internet Movie Script Database. `http://www.imsdb.com`. Accessed 04.02.2017.

[51] Apoorv Agarwal, Sriramkumar Balasubramanian, Jiehan Zheng, and Sarthak Dash. Parsing screenplays for extracting social networks from movies. *EACL 2014*, pages 50–58, 2014.

[52] Sebastian Gil, Laney Kuenzel, and Suen Caroline. Extraction and analysis of character interaction networks from plays and movies. Technical report, Stanford University, 2011.

[53] Fionn Murtagh, Adam Ganz, and Joe Reddington. New methods of analysis and semantics in support of interactivity. *Entertainment Computing*, pages 115–121, 2011.

[54] Syd Field. *Screenplay: The Foundations of Screenwriting.* Random House Publishing Group, 2005.

[55] Greg Beal. A few more days, February 2014. `https://www.oscars.org/sites/oscars/files/scriptsample.pdf`. Accessed 02.11.2016.

[56] US baby names. `https://www.kaggle.com/kaggle/us-baby-names`. Accessed 02.04.2016.

[57] Office for National Statistics - Live Births. `https://www.ons.gov.uk/peoplepopulationandcommunity/birthsdeathsandmarriages/livebirths`. Accessed 13.03.2017.

[58] IMDbPY. `http://imdbpy.sourceforge.net`. Accessed 10.05.2017.

[59] Gonzalo Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88, March 2001.

[60] Project Gutenberg. `http://www.gutenberg.org/`. Accessed 02.02.2017.

[61] Julian Brooke, Adam Hammond, and Timothy Baldwin. Bootstrapped text-level named entity recognition for literature. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*, 2016.

[62] Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 492–501, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[63] Nickname and diminutive names lookup. `https://github.com/carltonnorthern/nickname-and-diminutive-names-lookup`. Accessed 20.05.2017.