Natural Language Processing of Movie Scripts to Analyse the Plot Content, Characters, and Their Evolution

STUDENT RESEARCH PAPER

Of the course of study computer science

at the Duale Hochschule Baden-Württemberg Stuttgart

by

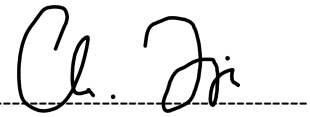
Deborah Djon and Lea Vergari

10.06.2022

Processing period	36 weeks
Matriculation number, course	9537809, TINF19A
	8521072, TINF19A
Dual Partner	Nokia Solutions and Networks GmbH Co. KG, Stuttgart
	Hewlett Packard Enterprise, Böblingen
Supervisor	Prof. Dr. Monika Kochanowski

Declaration of Originality

I hereby certify that I have written my student research paper with the topic: “Natural Language Processing of movie scripts to analyze the plot content, characters, and their evolution” independently and have not used any sources or aids other than those indicated. I also assure that the submitted electronic version corresponds to the printed version.

Stuttgart, 08.06.2022 

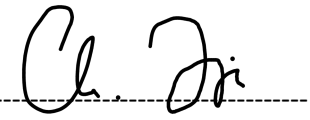
Place Date Signature

Stuttgart, 08.06.2022 

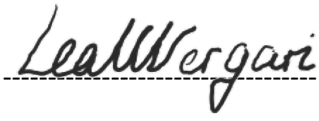
Place Date Signature

Erklärung der Eigenleistung

Ich versichere hiermit, dass ich meine Studienarbeit mit dem Thema: “Natural Language Processing of movie scripts to analyze the plot content, characters, and their evolution” selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich versichere zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Stuttgart, 08.06.2022 

Ort Datum Unterschrift

Stuttgart, 08.06.2022 

Ort Datum Unterschrift

Abstract

To make an informed decision about a movie or series, there needs to be an objective instance that can evaluate the movie or series. Therefore, the goal of this paper is the Natural Language Processing of movie and series scripts to analyse movie characters and their evolution.

First, the theoretical fundamentals of this paper are explained as a basis. To analyse the script, five separate hypotheses are then examined and from all these areas of interest, a method to generate an overview of an episode or season is developed to provide an objective overview. Interesting are the results of the first and the third hypothesis. The first one assumes that the main characters can be detected through the amount of speech and must be rejected since a main character has to speak to be detected as such. The third hypothesis assumes that men speak more than women and that these two genders discuss different topics. The first part can be accepted; however, the second part must be rejected as the men's, and women's distribution over the topics is ruffly the same.

There are some critical points, though. The work was executed with imperfect data and limited resources. Additionally, certain aspects of the technologies used have some drawbacks that need to be considered.

As an outlook, the points raised in the critical reflection can be optimized in future research. Furthermore, additional aspects like other characters or series can be inspected. In general, Natural Language Processing is a field that is still constantly evolving. Further developments can be expected and then integrated into this work.

Table of Contents

List of Abbreviations

BoW ..... Bag of Words

CRISP-DM ..... Cross-Industry Standard Process for Data Mining

DTM ..... Document Term Matrix

EDA ..... Exploratory Data Analysis

IMSDB ..... Internet Movie Script Data Base

LSA ..... Latent Semantic Analysis

NER ..... Named Entity Recognition

NLG ..... Natural Language Generation

NLP ..... Natural Language Processing

NLU ..... Natural Language Understanding

NMF ..... Non-Negative Matrix Factorization

NPMI ..... Normalized Pointwise Mutual Information

PDA ..... Personal Digital Assistent

PLSA ..... Probabilistic Latent Semantic Analysis

PoS ..... Part of Speech

SA ..... Sentiment Analysis

SVD ..... Single Value Decomposition

URL ..... Uniform Resource Locator

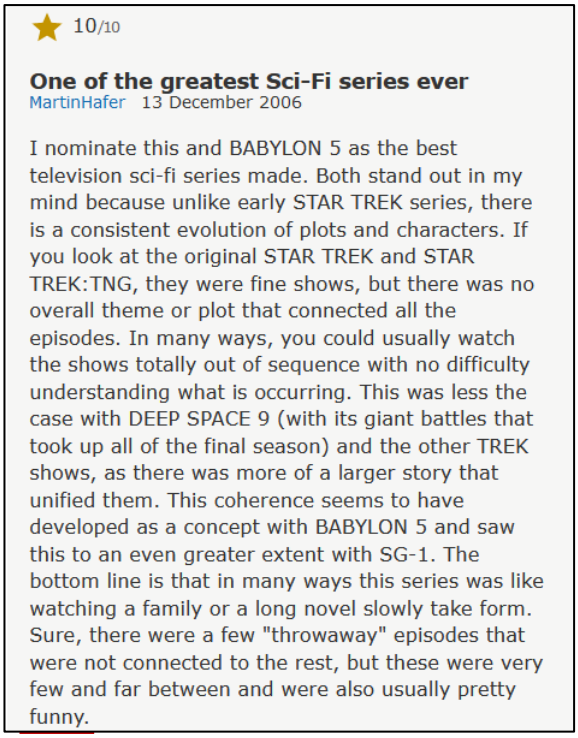
Table of Figures

List of Tables

1. Introduction

Numerous reviews and various summaries of movies and series are freely available on the Internet. The website IMDb alone has over 8 million titles listed, under which users can leave a rating or review [1]. The best-rated movie on the website alone had 10.027 reviews in March 2022 [2]. These reviews are dependent on the conscious and unconscious opinions of the authors, so summaries and reviews can often differ from the actual content of the film or series.

The TV series “Stargate SG-1” that will be analysed in this paper has a total of 263 reviews on IMDb [3]. The following Figure 1 shows an excerpt of the review with the highest rating as an example.


10/10

One of the greatest Sci-Fi series ever

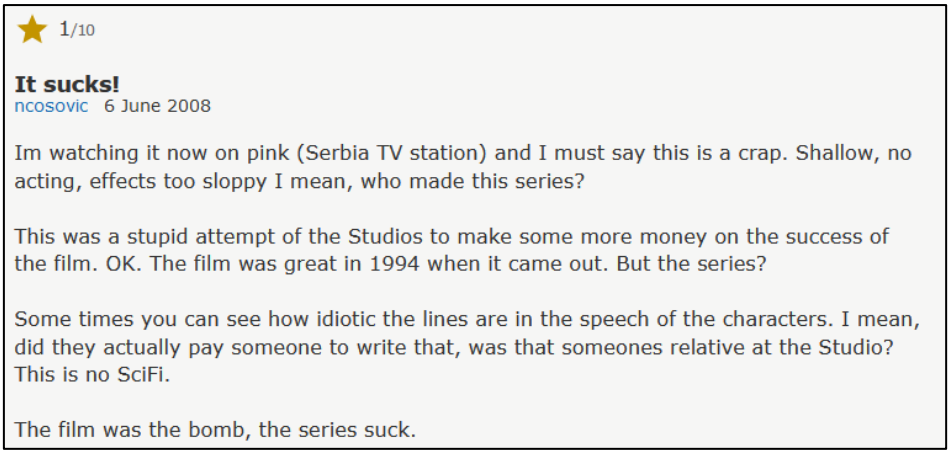
MartinHafer 13 December 2006

I nominate this and BABYLON 5 as the best television sci-fi series made. Both stand out in my mind because unlike early STAR TREK series, there is a consistent evolution of plots and characters. If you look at the original STAR TREK and STAR TREK:TNG, they were fine shows, but there was no overall theme or plot that connected all the episodes. In many ways, you could usually watch the shows totally out of sequence with no difficulty understanding what is occurring. This was less the case with DEEP SPACE 9 (with its giant battles that took up all of the final season) and the other TREK shows, as there was more of a larger story that unified them. This coherence seems to have developed as a concept with BABYLON 5 and saw this to an even greater extent with SG-1. The bottom line is that in many ways this series was like watching a family or a long novel slowly take form. Sure, there were a few "throwaway" episodes that were not connected to the rest, but these were very few and far between and were also usually pretty funny.

Figure 1: Excerpt of positive series review [4]

You can see in this example that the review is highly subjective and does not necessarily have profound, objective significance. For example, the author compares the series with other shows that are, according to the author’s opinion, comparable. Labelling some episodes as “throwaway episodes” is also highly subjective, same as considering the episodes as “pretty funny”.

In contrast to this review, Figure 2 shows a negative review of the series.



1/10

It sucks!
ncosovic 6 June 2008

Im watching it now on pink (Serbia TV station) and I must say this is a crap. Shallow, no acting, effects too sloppy I mean, who made this series?

This was a stupid attempt of the Studios to make some more money on the success of the film. OK. The film was great in 1994 when it came out. But the series?

Some times you can see how idiotic the lines are in the speech of the characters. I mean, did they actually pay someone to write that, was that someones relative at the Studio? This is no SciFi.

The film was the bomb, the series suck.

Figure 2: Negative series review [5]

The dialogue that was described as “funny” in Figure 1, is described as “idiotic” in Figure 2, and the story and development that was praised in the previous review is here described as “shallow”.

The two reviews prove that two people can watch the same movie or series and write reviews that are contractionary, as the reviews are based on subjective opinions.

Therefore, there is no instance that can provide a neutral view of a movie or series. An automatic summery generation would carry the risk of spoilers as the computer can not identify plot points that should not be revealed to the audience. It also gives no information about characters or the dialogue distributed among them. However, a way to create a fact-based analysis, is to have a computer assess the movie or series based on real facts that belong to the movie or series.

2. Problem Statement

In order to gain a neutral, content-based insight into a film or series without having seen the film or series in its entirety, there must be an objective way to analyse and, if necessary, evaluate the film or series.

The aim of this work is Natural Language Processing (NLP) of movie and series scripts to neutrally analyse the content (plot), the contained characters and their development. Analysing footage or audio is out of the scope of this project.

Therefore, we formulate five separate hypotheses:

1. First, it is assumed that one can tell if a character is a main character by the amount of speech they have.
2. The next hypothesis poses the assumption that the majority of the topics discussed are topics in the area of science fiction.
3. The third hypothesis focuses on the differences between male and female characters in two different ways. The first assumption is, that male characters have a higher proportion of speech than women. Second, it is assumed that men and women talk about different topics.
4. As the fourth hypothesis, we assume that the amount of speech of individual characters changes significantly across seasons.
5. Finally, the last hypothesis assumes that the sentiment of the characters tends to be neutral, based on the assumption that there are no strong opinions in the plot as there would be, for example, in reviews.

Based on those hypotheses, the goal is to compose a short overview for an episode or season that consists of the characters featuring the episode or season, the three characters with the biggest amount of speech, how many words of dialogue the episode or season contains, the topics that are mentioned, a graph showing the sentiment over the duration of the episode or season and a mean value, and a timeline that shows the amount of speech of characters within the season.

3. Theoretical Fundamentals

The project is concerned with Natural Language Processing (NLP). The following outlines the theoretical fundamentals for the following chapters. First, the fundamentals of Natural Language Processing are outlined. Afterwards, the Natural Language Processing concepts and methods used in this project are described. These are firstly *Text Pre-Processing Steps in Natural* Language Processing, then *Topic Modelling* and lastly *Sentiment Analysis*.

3.1 Natural Language Processing

Natural Language Processing has emerged from different fields such as artificial intelligence, linguistics, formal languages, and compilers and thus overlaps with many areas [6, 7]. A more detailed definition and possible application areas are discussed in the following subchapters.

3.1.1 Definition

Natural Language Processing (NLP) is concerned with deriving insights from and producing samples of natural language. The former is referred to as Natural Language Understanding (NLU), and the latter is referred to as Natural Language Generation (NLG). Natural language is any text or audio that was produced by a human [6].

More formally, one can define Natural Language as a “mutually agreed set of protocols involving word/sounds we use to communicate with each other” [6]. It can be considered as unstructured data and thus has to be pre-processed and formatted to be usable for computers [8].

With Natural Language Processing, natural language is first transformed into a machine-readable format. Once the data is transformed, Natural Language Processing techniques such as Sentiment Analysis or Topic Extraction can be applied [6].

The goal of Natural Language Processing is to enable human-computer interaction via natural language. Historically, Natural Language Processing used rule-based systems to analyse text; however, today, most computations are based on machine learning [6, 7].

3.1.2 Applications

Natural Language Processing is applicable in various application areas. Example areas that are discussed here are machine translation, text categorization, spam filtering, information extraction, summarization, dialogue systems and medicine [7].

Machine Translation

A major problem with data accessibility is the language barrier. Machine translation is used to translate a sentence from one language to another and has a history going back over 60 years. The challenge is not directly translating the words but preserving the meaning of the sentence while using correct grammar and tenses. As there are numerous languages with different sentence structures and grammar, this can be difficult [7, 9].

Thus, Natural Language Processing is used in different approaches, for example, a statistical machine learning approach that collects data that seems parallel between two languages and then calculates the likelihood that a word in one language corresponds to another [7].

In recent years many approaches, like using artificial neural networks, or comparing hypothesis translations with reference data, have been developed. There are different criteria to assess the translations. With them, an approach can achieve a good evaluation that correlates with a human assessment [7].

Text Categorization

As the name suggests, text categorization uses Natural Language Processing to sort and filter big amounts of data into different categories. The technology has seen a rise in interest over the last

years due to the rising amount of data that is digitally available. With acquiring more data comes the need to organize it automatically. Using text categorization saves time by having a computer doing the task instead of a staff member [7, 10].

Text categorization is used in many contexts, such as document indexing that is based on a controlled vocabulary, document filtering, automated metadata generation, word sense disambiguation or generally applications where (automatical) document organization is necessary [10].

Example applications that use text categorization are systems like trouble ticket categorization and spam filters [7].

A lot of text categorization algorithms use supervised machine learning to label the incoming data with the associated category using a model that was trained with pre-labelled datasets [11].

However, especially in spam filters that are widely used, there are various other possible approaches such as *content filters* that review the content of data to select a label, *header filters* that regard, for example, the e-mail header to decide, *general blacklist filters*, that in the special case of spam filters – have a list with recipients that are automatically considered spam; *rule-based filters* that use user-defined rules to decide, *permission filters* where a sender needs explicit permission from the recipient and lastly *challenge response filters* that require a sender to enter a code to gain the permission to send the message [7].

Information Extraction

Information extraction is used to identify phrases of interest in texts of natural language. This can be extracting entities such as names, places, events or dates. Processing steps that are used within Information Extraction can be Part of Speech tagging (PoS), stemming or word sense disambiguation. These techniques will be further discussed and explained in subchapter 3.2 “Text Pre-Processing Steps in Natural Language Processing” [7].

Information extraction can be used in many different ways. As an example, it can be used to build a database, prepare summaries, classify text items into predefined categories or identify keywords [7].

It is suggested that information extraction performs well when extracting terms or phrases from documents, however determining relationships between different terms is still a challenge and needs to be further developed [7].

Automatic Summarization

The goal of automatic summarization is to automatically determine and extract important content from a source of information, for example, a document, and present it to the user in a way that fits the required format according to the application or usage. Important to note is that the size of the summarization is significantly less than the original source [12].

In the context of Natural Language Processing, we talk about summarizations in text form. In that sense, this specification is important, as a text document could also be summarized in different ways, lika a graphic or a trailer [12].

Automatic summarization applications can be categorized into either single-document summarization or multi-document summarization. Depending on which one applies, different approaches for summarization are used. Summarizers can work supervised or unsupervised. In the case of a supervised summarize technique, a big amount of labelled data is needed to train the model [7].

While thus work also aims to generate an overiew over a series, there was a conscious decision against using automatic summarization. As the text that is supposed to be summarized consists only of thousands of lines od speech, there is no contignuos text that can be summazied.

Dialogue Systems

A dialogue system is a computerized system that communicates with a human user by using natural language. In the future, this can enable robots to interact with humans naturally. The focus lies in devices like cars, *Personal Digital Assistents* (PDA), or even smart refrigerators. Existing examples of software and devices that are dialogue systems are the Google Assistant, Alexa by Amazon, Siri by Apple and Cortana by Windows [7, 13].

Dialogue systems usually provide a user interface that serves as a bridge between the human user and the computer system. Generally, a dialogue system can use text-based, spoken or multimodal dialogue [13].

A text-based system uses a chat to interact with a user, while a spoken dialogue system uses spoken natural language on a turn-by-turn basis. A multimodal system combines two or more ways of user input. Possible input ways could be speech, touch, manual gestures, gaze, pen, and head and body movements [13].

3.2 Text Pre-Processing Steps in Natural Language Processing

There are many possible data pre-processing steps in Natural Language Processing. The following outlines common steps when pre-processing data for Natural Language Processing projects. Not all of those were used to process the data for this specific project of analysing series scripts.

3.2.1 Tokenization

Tokenization is the process of splitting a sentence into the parts building the sentence. English is a space-delimited language, meaning that words are separated by blank space. In contrast to, for example, Chinese or Thai, the tokenizer separates the words by blank space, thus creating so-called tokens [14]. In this case, the tokenization would create so-called *unigrams*, as each token consists of only a single word. Tokens can also consist of more than one word and are then labelled as *bigrams* (consisting of two words), *trigrams* (consisting of three words) or generally *n-grams*, where *n* is a natural number, and the term describes a token build of n words [6].

Tokenization can be difficult when a number is specified that describes the length of tokens. For example, when unigrams are used, there might be terms like for example, names that consist of more than one word and separating the words would change or remove the meaning of that term. An example of this would be ‘United States’. Separating the two words would altogether remove the fact that this is the name of a country [6].

3.2.2 Part of Speech Tagging

Part of Speech (PoS) tagging labels words as their part of speech, for example, verb, noun or adjective [14].

It is used to label the words in a sentence. Then the labels are used to filter out the part of speech of interest for analysis. The labels themselves describe which grammatical word class the word belongs to. Next to the actual part of speech label, a part of speech tagger usually produces additional information like inflectional and lexico-semantic information, for example, a differentiation between a common and a proper noun [15]. It is most often used on unigrams that were generated through tokenization. Thus, tokenization is normally a processing step that has to precede Part of Speech tagging [6].

3.2.3 Stop Word Removal

Stop words are words that need to be included in a sentence to make it grammatically correct and readable. Example stop words could be ‘a, ‘am,’ or ‘the’. These words appear very often; however, they add no real meaning to the sentence and can thus be removed without impacting the meaning of the sentences. For certain examinations of the text, this step is not optional as it would distort the result. An examination where this would be applied could be the examination of which words are used the most in a text or the creation of word clouds based on the occurrence of a word [6].

Stop words are often removed by using existing stop word libraries. The text is tokenized, and the tokens are then compared with the words on the list. Tokens that equal a stop word are removed [6].

3.2.4 Text Normalization

Text normalization is the process of converting words into their standard forms. There are words that can be spelt, pronounced, or represented differently but mean the same thing. This can be inflected verbs or different versions to write a word like, for example, 'US' and 'United Stated' or simply spelling mistakes. Normalizing text can happen through different processes like spelling correction, stemming and lemmatization [6]. These will be discussed in the following.

Spelling Correction

As the term implies, spelling correction includes the removal of spelling mistakes in a word. There are various libraries that can be used to detect mistakes and automatically correct them [6].

Generally, there are two ways to correct the words. One can either use a lexical approach that compares words with the lexical spelling or use a context-based approach. The second option has the advantage of also detecting mistakes that turn a word into another, for example, 'on' turning to 'in' [16].

Stemming

Due to grammatic rules, such as using plural forms or inflecting verbs, words change their spelling while the meaning stays the same. To help with the analysis process, these words need to be converted back into their basic form [6].

Stemming is the process of converting a word into its *stem* [14]. The aim is to reduce inflexion forms and related forms of a word to its stem by usually removing suffixes and prefixes. Stemming is especially important for indexing and search systems [17].

With stemming, there can be two general mistakes: over-stemming and under-stemming. Over-stemming describes the process of stemming two words to the same root when they should be different roots, and under-stemming happens when two words should have the same root but are stemmed to different ones [17].

Lemmatization

Lemmatization is very similar to stemming. It also has the goal of transforming a word into a basic form, the so-called *lemma*. However, it uses a different approach. While stemming only reduces a word with certain rules, lemmatization adds a step by including the part of speech into the process. Lemmatization does an additional inspection with a dictionary to understand the part of speech and context of the word and verify the reduced form. However, this additional step makes the process slower compared to stemming [6, 17].

3.2.5 Named Entity Recognition

Named Entity recognition (NER) aims to identify named entities like proper nouns, for example names of places or people. They are usually not included in dictionaries and thus have to be handled separately [6]. Named Entity Recognition is the process of locating and categorizing these words [18].

The process of Named Entity Recognition usually uses so-called *gazetteers* that are basically lists of names of people, locations, organizations, or other named entities. Named Entity Recognition is especially important for tasks such as information extraction or retrieval, where it can be used to differentiate between names and general words. For example, it could be used to search for a person with the name “Jobs” without the system retrieving documents bout actual jobs [19].

3.2.6 Word Sense Disambiguation

Word Sense Disambiguation handles the problem of several words with the same spelling having different meanings depending on the context. This often causes disambiguation. Therefore, it is the process of mapping a word to its correct meaning in the specific context of the sentence [6].

This is important for categorizing them as entities or Part of Speech and analysing them accordingly. An example could be the word ‘play’. According to the context, this could be a verb that describes playing a game, or it could be a noun in the sense of a theatre piece. Additionally,

it could also be a name for a book. Depending on what is analysed in a specific project, it might be important to have this information about the sense of the word [6].

3.3 Topic Modelling

Topic modelling is a Natural Language Processing (NLP) technique for extracting general themes and ideas from a set of texts. Topic modelling is a form of dimensionality reduction. It reduces texts to a finite number of themes, called topics. Modelling topics is challenging as the data is mostly unlabelled, and the number of topics is not defined. To overcome these challenges, topic modelling uses unsupervised machine learning algorithms, which often work with unlabeled data and aim to group information. Popular topic modelling algorithms and tools include Latent Semantic Analysis (LSA), Latent Dirichlet Allocation (LDA), Term Frequency Inverse Document Frequency (TFIDF), and Non-Negative Matrix Factorization (NMF). The following sections outline these algorithms. They further describe how topic modelling algorithms are evaluated and improved [6].

3.3.1 Terminology

Certain terms are reoccurring in most topic modelling projects. The following paragraphs describe key topic modelling terms that are used in this paper.

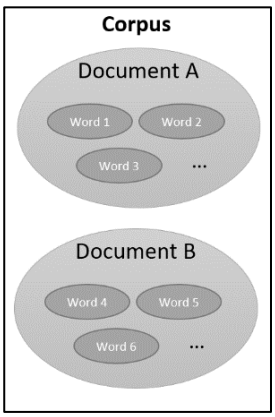


Figure 3: Relationship corpus, documents, and words [6]

A *document* describes a unit of text. It contains multiple *words*. Examples of documents are tweets, articles or books. Multiple documents are confined in a so-called *corpus*. Figure 3 describes the relationship between these terms. All words of a corpus can be tracked in a *dictionary*. It contains key-value pairs of all words (value) and a unique identifier (key) [6].

A *Bag of Words* (BoW) is a natural language data structure for converting textural input into numerical input. Figure 4 exemplarily shows how a

corpus is converted into a Bag of Words. The Bag of Words can be represented as a table where the columns represent all words present in a corpus. The rows represent all documents present in a corpus.

Corpus	
d_1	"Bill has a house"
d_2	"Bill and anna build a house"
d_4	"Anna loves bill."

Bag of Words								
	bill	has	a	house	and	anna	build	loves
d_1	1	1	1	1	0	0	0	0
d_2	1	0	1	1	1	1	1	0
d_3	1	0	0	0	0	1	0	1

Figure 4: Conversion corpus to Bag of Words [6]

The table contains numerical values that represent each word in each document. If the contents of the Bag of Words are the word frequencies, it is also referred to as a Document Term Matrix (DTM). However, the content of a Bag of Words can also be a normalized metric or other suited values like the *Term-Document Inverse Document frequency* (Tf-idf) metric. The Tf-idf is a measure that helps rank words based on how good they are at identifying documents.

$$tfidf = tf * \log\frac{N}{df}$$

Equation 1: TF-idf formula [6]

The Tf-idf formula shown in Equation 1 consists of two parts. The first part, the Term-Frequency, simply ranks all words by their overall occurrence in all documents. The second part, the Inverse-Document Frequency, helps punish words that occur in many documents. This way, only words that often occur but in few documents get a high Tf-idf score [6].

In topic modelling, a document consists of a set of *topics* and topics are a group of words [6].

3.3.2 Topic Modelling Techniques

This section describes the most popular topic modelling techniques, their strengths and their weaknesses.

Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a topic modelling technique for calculating similarities between documents and terms. In Latent Semantic Analysis, the documents and terms are first represented as a Document Term Matrix or Bag of Words. After creating the document-term matrix, the matrix is decomposed into a term-topic matrix and a document-topic matrix [6] .

A Document Term Matrix is sparse, as it contains many zeros. This sparsity can be counteracted by reducing the dimensionality of the matrix. In Latent Semantic Analysis, dimensionality reduction is made using Single Value Decomposition (SVD). This algebraic process decomposes a given matrix into the product of three matrices U, σ, and V. U describes the document-topic matrix, V is the term-topic matrix and σ describes the topic importance. As shown in Figure 5 the dimensionality reduction is performed by selecting only the t-biggest values from the matrices. The resulting document and term matrices can be used to determine the similarity between documents and terms. An example of a function for calculating the similarity is the cosine similarity.

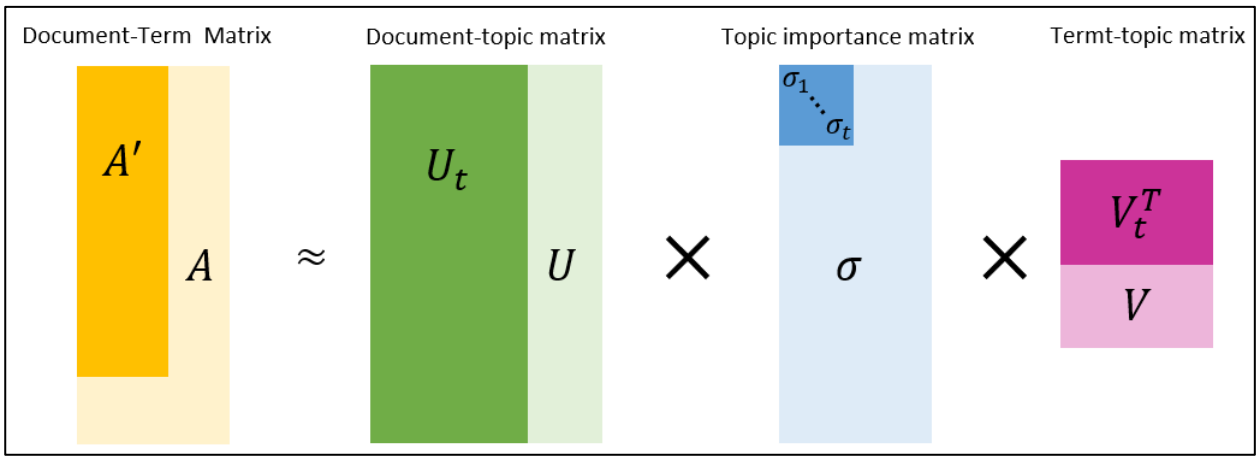


Figure 5: Single Value Decomposition Scheme [6]

Although LSA is an efficient and fast method for topic modelling, it has drawbacks. An example drawback is that LSA requires extensive data sets for accurate results. Further, the results are not interpretable since the topics are unknown [6] .

Probabilistic Latent Semantic Analysis (PLSA) is similar to LSA. However, instead of Single Value Decomposition, it uses a generative probabilistic approach. PLSA aims to create data that is similar to the given document-term matrix. PLSA is similar to LSA because there is a direct correlation between the parameters of the PLSA and the LSA equation. Figure 6 describes the relationship between the two formulas [20].

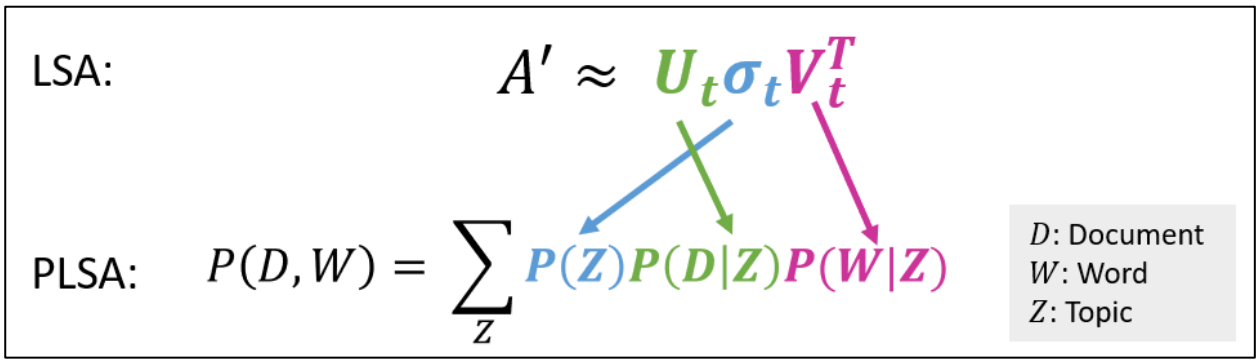


Figure 6: Comparing LSA and PLSA [20] [21]

The probabilistic approach of PLSA has the advantage that the model can be adjusted using standard statistical methods, like cross-validation, for measuring the predictive performance. On the other hand, PLSA has some shortcomings. As there are no parameters for modelling the probability of a document ($P(D)$), probabilities cannot be assigned to unseen documents. Further, PLSA does not scale well. The number of parameters used increases linearly in accordance with the document count [20].

Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a Natural Language Processing method for discovering themes in a group of texts. These themes are called *topics*. The method is also an unsupervised machine learning algorithm. This means no labels are needed to train the algorithm. In Latent Dirichlet Allocation topics are distributions over words. Also, documents are distributions over topics [6].

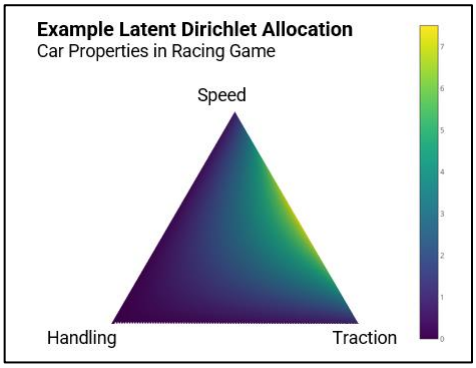


Figure 7: Latent Dirichlet Allocation Example [22]

LDA makes use of the so-called *Dirichlet distribution*. This is a continuous probability distribution that describes the probability distribution for a set of input elements. The input can be of any range. An example that can be modelled with a Dirichlet distribution are the car properties in a racing game. The user gets a fixed number of points and allocates them to the three properties speed, handling and traction. The probabilities of the properties all add up to 1. The example is visualized in **Fehler! Verweisquelle konnte nicht gefunden werden.**

The graphic shows that most users choose to allocate their points between speed and traction. In Latent Dirichlet Allocation, a topic is such a Dirichlet distribution over a set of words. Further, each document is assigned a Dirichlet distribution over the found topics. LDA tries to find these latent or "hidden" Dirichlet distributions behind topics and documents [22, 23].

LDA is a generative probabilistic model. It internally generates several documents that correlate to a defined number of topics. Through an iterative process called Gibb's Sampling, LDA adjusts the topic model until the generated documents closely match the original documents [22].

LDA is the most popular topic model and is highly studied. Therefore, Python offers many libraries and tools for LDA topic modelling as well as evaluation tools. Further, LDA addresses some of the shortcomings of previously described topic models. The topics are interpretable, they can be applied to new documents, and LDA does not require very large amounts of data [22][24].

Non-Negative Matrix Factorization

Non-Negative Matrix Factorization (NMF) is an algebraic method for dimensionality reduction. The input to the method is a non-negative matrix A. Here, non-negative means all entries are greater or equal to zero. The output is a matrix multiplication of two matrices, W and H. The columns of W and the rows of H are of size k. The k represents the relationship between W and H. Non-Negative Matrix Factorization can be used for topic modelling. In this case, the input matrix is the Document Term Matrix with the number of rows "n" representing the terms and

the number of columns "m" representing the documents. The matrix factorization then decomposes the matrices into a term-topic matrix W and a topic-document matrix H. k represents the topics. Figure 8 describes the NMF decomposition [24].

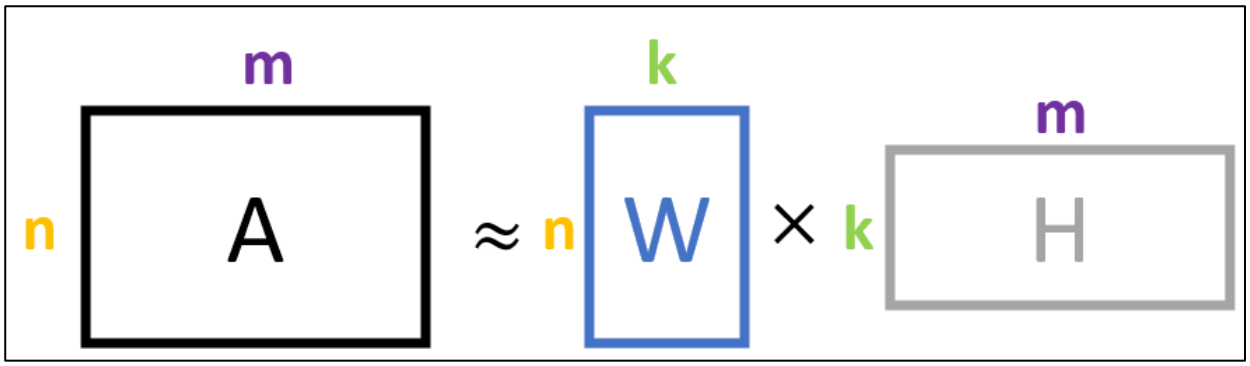


Figure 8: Non-Negative Matrix Factorization Schema [24]

NMF is inherently a minimization problem that is not proven to be solvable analytically. Hence, it must be approximated numerically. Consequently, the result of a Non-negative Matrix Factorisation is not unique and can change with new iteration cycles. NMF is preferred for shorter texts such as tweets, questions or search queries [24]. Further, NMF has been shown to perform faster than methods like LDA [25].

3.3.3 Evaluating Topic Models

As mentioned at the beginning of this chapter, topic modelling uses unsupervised machine learning techniques. This makes evaluating topic models challenging as there is no binary metric for labelling an outcome correct or incorrect. Nonetheless, there are methods for determining the quality of a topic model. Such methods include perplexity, word intrusion, topic intrusion, and coherence metrics. Further, topic models can be evaluated through visualizations like word clouds or the python pyLDAvis library. The following outlines popular measures for evaluating topic models [26][27][28].

Perplexity

A common method to evaluate topic models is to compare a model to held-out data that is not involved in the training process. In a generative language model such as Latent Dirichlet Allocation, the model is a set of words that are created by the algorithm. These words define a topic. The topic model is good if the words in a topic do not differ much from the words in the the documents assigned with the topic. Perplexity is a metric for measuring the difference between a language model and actual data. A high perplexity score indicates a low-quality topic model. It means that the model differs largely from the held-out data. With held-out perplexity, a topic model is evaluated against a held-out set of documents that were not used in the training process [29].

Chang et al. [30] describe held out perplexity to have a high predictive capability. However, the models lack interpretability. This is problematic as topic models are used in contexts where humans must understand the topics. An example is a newspaper publisher that tries to understand the main themes in recent articles. Therefore, it is not beneficial for a topic model to correctly predict topics if a human cannot interpret the topics [30].

Word and Topic Intrusion

Chang et al. [30] propose another method for evaluating topic models. They offer to incorporate human judgment in the evaluation process. The method they introduce is *word intrusion*. With this method, several words belonging to a topic are presented to a human. Another word that does not belong to the topic is added. If the human can determine what word does not belong to the topic, the model has a high quality. As mentioned, this method is preferable, as the topics are interpretable. Chang et al.'s finding show that the perplexity measure correlates negatively with the word intrusion metric [30]. This weakens the meaningfulness of the perplexety measure, as topic model evaluation metrics that are based on human judgement are the gold standard. However, these evaluation metrics are expensive as they require many humans to gain results with significant statistical power [30].

Coherence Metrics

Other metrics for topic models are coherence measures. A set of statements is coherent if they are related to each other and make sense together. Coherence metrics compare words in a document and see how often these words occur together in external texts. An example of an external text is Wikipedia. Coherent metrics can be preferable over the word intrusion method because they are less expensive [30].

Röder et al. [27] compared a set of coherence metrics. They found that measures using the C_v score performed the best out of all regarded measures. However, the C_v score has flaws. It behaves badly with randomly generated sets of words. Therefore, it is also not recommended by the author [31]. Other also well-performing coherence scores are the C_p and the Normalized Pointwise Mutual Information (NPMI) score.

However, recent findings show that past research regarding automatic topic model evaluation is flawed. The researchers Hoyle et al. discovered that the metrics used for comparing and evaluating the metrics, such as the C_v score are not very meaningful. Mainly the NPMI score. There are neural models that have much higher NPMI scores. However, it is unclear if they perform better for several reasons. One being that there is no standardized NPMI formula. Another reason is that the neural models were seldom validated against human coherence preferences. Automated coherence metrics are still useful, for ranking topic models. However, for determining the objective quality of a topic model, further research is needed.[32].

Visualizations

When a topic model is created, it can be inspected by means of visualization. This can help as a secondary measure to evaluate the meaningfulness of the model. An example is shown in Attachment A**Fehler! Verweisquelle konnte nicht gefunden werden.**. It shows the visualisation o f an LDA model. The visualisation is created with the python LDAvis module.

The module creates a two-part visualization. On the left, it shows the topics and their distances using a bubble chart. The site of the bubbles corresponds to the prevalence of the topic. When

selecting a topic, the right side of the visualization shows the 30 most important words in a topic. The length of one blue bar describes the frequency of the word in the topic. The red part of a bar shows the relative frequency of a word in the selected topic [28].

3.3.4 Improving Topic Models

As stated in the introduction of this chapter, topic modelling uses unsupervised machine learning algorithms. The base machine learning model is mainly improved with hyper parameter tuning. Hyper parameters are the variables a model is trained on. In topic modelling, an example of a hyper parameter is the topic count. The most common method for determining the best hyper parameters of machine learning models is *grid search*. This method tries out all possible combinations of hyper parameters, and then the model with the highest value of the evaluation metric is chosen [33].

3.4 Sentiment Analysis

Sentiment Analysis (SA), also called opinion mining, is defined as the computational study of an entity that can be an individual, event, or topic. The goal is it here to determine the opinion of a text's author as positive, negative, or neutral [34, 35].

Sentiment Analysis is currently in the focus of research, as this technic offers organization insights into customer reviews. There are over 7000 articles published on the topic, and many start-ups develop technologies that implement Sentiment Analysis [34].

Since different social media networks like Facebook or Twitter offer a platform to publish one's opinion, Sentiment Analysis helps businesses to monitor the public’s or their customer's opinions in real-time. Marketing managers, PR firms, politicians, and online shoppers benefit directly from Sentiment Analysis, but it is also used in the financial market, where blogs and forums are

analysed to find a general sentiment towards a company. With a negative sentiment, the stock price is likely to sink, while it is likely to increase with a positive sentiment [34].

A sentence can be classified into two categories: objective or subjective. An objective sentence contains facts, while a subjective sentence expresses an opinion or belief about an entity. Sentiment Analysis concentrates mostly on subjective sentences but can also be used in objective contexts [34, 35].

A text that is analysed can include different topics that the author has different opinions about. For example, a movie review can have opinions about the acting, story, and special effects, that the author can have different thoughts about. They could, for example, express a positive view about the story but criticize poor acting and bad special effects. A good Sentiment Analysis should be able to pick up the different sentiments about the three subtopics, as well as analyse the general sentiment about the movie [34].

3.4.1 Approaches for the Usage of Sentiment Analysis

As a generic architecture, one can say that the Sentiment Analysis system receives a corpus of documents as input that can have any format, for example, PDF or XML. These documents are then converted to text and pre-processed using tools like stemming, tokenization, Part of Speech tagging, entity extraction, and relation extraction [34].

Entity extraction determines which entities are addressed in the text. The relation extraction searches for the relationship between different words [14].

After the pre-processing of the text, the document analysis module uses linguistic resources like lexicons to label the document, sentences, or aspects with the corresponding sentiment. As there are different degrees on how deep a Sentiment Analysis can go into detail, there are five different approaches to how Sentiment Analysis can be used: Document-level Sentiment Analysis, sentence-level Sentiment Analysis, aspect-based Sentiment Analysis, comparative Sentiment Analysis and sentiment lexicon acquisition [34].

Document-level Sentiment Analysis

Document-level Sentiment Analysis is the simplest approach to Sentiment Analysis and aims to classify the sentiment or opinion on the whole text. To return to the previously mentioned example – the movie review –in this approach, the goal of the Sentiment Analysis is classifying the author's general opinion about the movie [34, 35].

This level of inspection assumes that the opinion in the text is about a single entity and expressed by a single author. If this is not the case, a different approach is needed. To date, most document-level techniques are based on supervised learning approaches, however, unsupervised methods exist [14]. The different techniques will be addressed in subchapter 3.4.2 "Sentiment Classification Techniques".

This project uses document-level Sentiment Analysis, as each script line is considered a document.

Sentence-level Sentiment Analysis

The sentence-level Sentiment Analysis inspects each sentence and determines the expressed sentiment. This approach is used when a more specific view about the entities is needed and/or one document contains opinions about several entities. In this approach, it is assumed that the entity that is discussed in a sentence is known and that the sentence contains only a single opinion on this entity. The latter part can be softened by splitting a sentence into several phrases that each contain one opinion [34].

Each sentence is first classified as objective or subjective and in the case of a subjective sentence, is examined to assign either a positive or negative sentiment [35]. Depending on the task, a third option, "neutral" can be added.

Sentence-level Sentiment Analysis is most often based on supervised or unsupervised learning. However, research has shown that different types of sentences require different strategies. Special types of sentences are for example, question sentences, conditional sentences and sarcastic sentences [34].

Aspect-based Sentiment Analysis

The aspect-based Sentiment Analysis, or feature-based Sentiment Analysis, provides the most details about a document. This approach's aim is to classify the author's opinion about several aspects (features) of an entity. The first step, before the Sentiment Analysis can be executed, is to identify the aspects that are included in a sentence. Like this, different opinions about different aspects of the same entity can be found [34, 35]. An example would be a movie review that mentions the storyline, acting and special effects, "*The movie had great special effects and acting, but the storyline was rather flat.*" In this example, the Sentiment Analysis would discover the positive sentiments towards special effects and acting but a negative sentiment towards the story [35].

This mostly happens in reviews of products or topic-specific discussion forums. Thus, the aspect-level Sentiment Analysis is most often used by commercial companies or marketing departments [34].

Comparative Sentiment Analysis

Comparative Sentiment Analysis is a subset of Sentiment Analysis. Often, the author of a review does not directly express an opinion but offers a comparison to another entity, for example, a product, and rates one as better or worse than the other [34].

In this case, it is the analysis' aim to discover sentences that offer such a comparison and identify the entity that is preferred [34].

Inspecting the sentences for comparative adjectives/adverbs like "*less*", "*more*", or words ending on *-er*, for example "*brighter*"; superlative words like "*least*", "*most*" or words ending on *-est*, for example "*greatest*", and additional phrases such as "*prefer*", "*superior*" or "*than*", can already find 98% of opinionated sentences [34].

3.4.2 Sentiment Classification Techniques

Generally, there are three different techniques to classify the sentiment: the machine learning approach, the lexicon approach, and a hybrid approach.

An overview of them is shown in Figure 9.

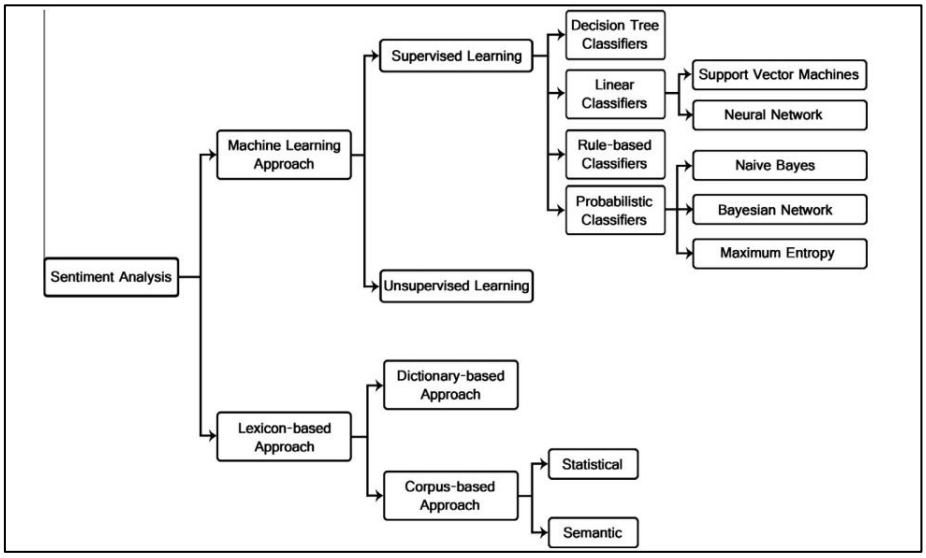


Figure 9: Sentiment classification techniques [36]

Machine Leaning uses supervised or unsupervised learning to train a model that can classify the text into the categories positive, neutral and if necessary neutral sentiment.

Thereby, supervised learning relies on a dataset of labelled training documents [35].

There are various classifiers for supervised learning but some that are frequently used in Sentiment Analysis are probabilistic classifiers, linear classifiers or decision tree classifiers as shown in Figure 9 [35].

However, it is often difficult to acquire a big enough amount of labelled documents. In this case, unsupervised training can be used. This approach uses linear discriminant analysis to find global topics with multiple aspects. Then, local topics are extracted with the associated sentiment on a sliding window context over the training text. Supervised learning is not used often in the classic Sentiment Analysis approaches [34, 35].

By using a lexicon, the algorithm searches the text for opinion words, phrases, and idioms in the lexicon [35]. The sentiment lexicon is the most important resource for Sentiment Analysis , as it assigns a word with the corresponding sentiment and is used to determine the sentiment of a document, sentence, or aspect [34].

There are generally three possibilities to create a sentiment lexicon: manually, dictionary-based or corpus-based [34].

The manual approach is the most arduous one, where the lexicon is programmed manually. For the dictionary-based approach, a small set of seed sentiment words starts as the core and is then expanded by using synonyms and antonyms. The disadvantage of the dictionary-based approach is that the resulting lexicon is domain independent, thus not including domain specific peculiarities [34].

To create a domain specific lexicon, the corpus-based approach must be used. A popular approach is to start with a set of words where the sentiment is already known. This set is then expanded by adding words that are concatenated to the known word with words like "and" or "or". For example, if a review describes an entity with two words that are connected with an "and" and the first adjective is known to be positive, then it can be assumed that the second adjective is also positive [34].

A hybrid approach combines the previous two approaches [35].

3.4.3 Research Issues

There are several open issues that still have to be researched. The most prominent ones are described here [34].

The first one is the necessary enhancement of compositional sentiment on for example sentence level. The parts of the sentence need to be analysed for their sentiment before calculating a general sentiment for the sentence. This calculation needs to be more accurate. The same problem applies to document-level approaches [34].

Another aspect that needs to be resolved is automatic entity resolution. An entity that is supposed to be assigned a sentiment can have different names. The detection of these entity names still has to be researched [34].

Related to the previous problem, it is also important that, in a text that contains several entities, these entities are cleanly separated, and it is identified which parts of which sentences are relevant for each entity [34].

Furthermore, detection of sarcasm is not yet integrated in most Sentiment Analysis systems and so called 'noisy text' that contains spelling or grammatical mistakes or uses slang words, is also difficult to analyse for current Sentiment Analysis systems [34].

Lastly, the current approaches for Sentiment Analysis rely heavily on subjective sentences about entities. However, objective sentences can also carry sentiment towards the entity, but are disregarded in current approaches of Sentiment Analysis systems. Thus, there is a need for approaches that use context to assign a sentiment to objective sentences like they are often used in for example news articles [34].

3.5 State of the Art

Works and development done in the field of Natural Language Processing can be dated back to the late 1940s. Generally, the development of Natural Language Processing can be organized into four phases: the late 1940s to late 1960s, the late 1960s to late 1970s, from the late 1970s to the late 1980s and the fourth phase from the late 1980s to now [37].

The first research evolving around Natural Language Processing in the late 1940s dealt with machine translation. One of the first projects in this time was conducted by Weaver and Booth, who developed computer translation based on knowledge from breaking cryptographic codes in 1946. Weaver's memorandum on translation in 1949, however, sparked the actual research around Natural Language Processing in the early 1950s [37, 38].

Research first assumed that the difference between languages only consisted of the different vocabularies and the allowed order of words. Thus, the technic of looking up translations in

dictionaries was used. This led to poor results due to the lexical ambiguity. In 1957 Chomsky published ideas of generative grammar that helped machine translation. Further, his findings led to other applications of Natural Language Processing being developed, such as speech recognition [38].

Those were the beginnings that were further developed in the 1960s. Research mainly focused on new computational solutions to represent meaning and model grammar. That phase used artificial intelligence and put more focus on word knowledge and the resulting role in constructing sentences. It included several prototypes, like ELIZA, a programme by Weizenbaum that chatted with a user by echoing the user's input [37, 38].

The third phase, from the late 1970s to the late 1980s, used grammatical theory to solve Natural Language Processing methods. Several types of grammar were developed to model natural language that was also relevant in parsing methods. This led to fundamental works around Natural Language Generation, like TEXT, a programme by McKeown that created comprehensible responses online [37, 38].

In the current phase, which began in the late 1980s, research moved to statistical language data processing. Information extraction and automatic summarisation is another focus that evolved [37, 38].

Natural Language Processing is long since used in various application areas such as machine translation, spam filtering or dialogue systems [7].

The following describes a current example application from the field of Natural Language Processing.

The project we want to discuss is "Eno", a chatbot by the US bank "Capital One" that uses natural language to interact with customers via SMS. Eno was developed in 2017 after customers started answering to automated fraud alerts via SMS [39].

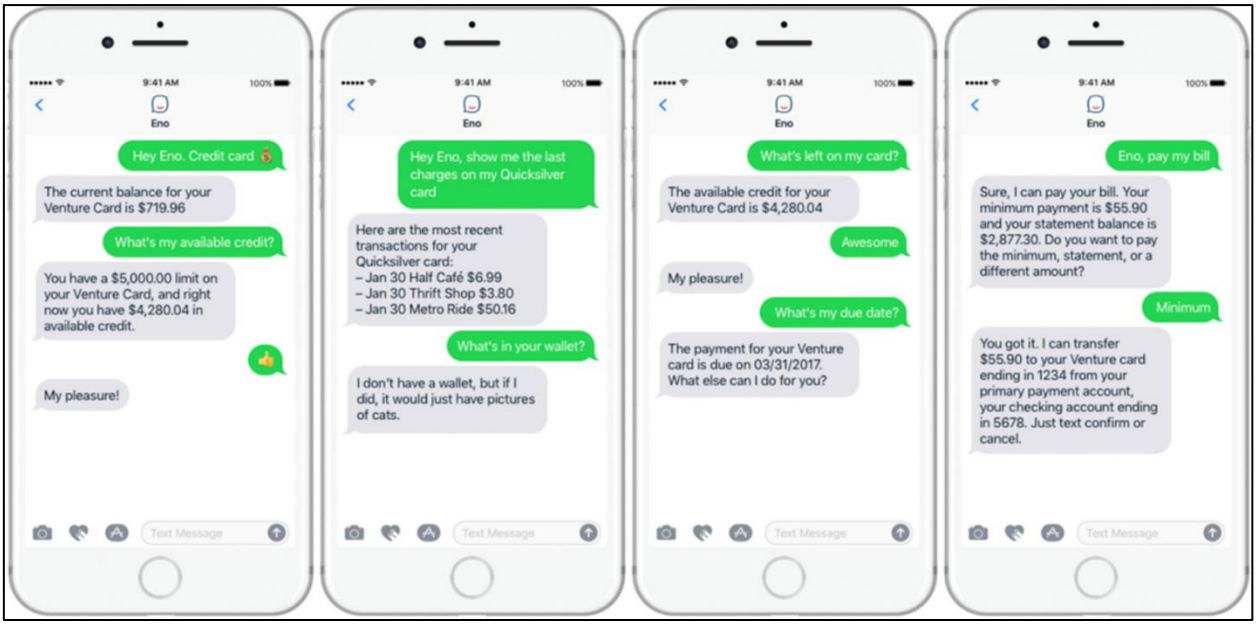


Figure 10: Example for interaction with Eno [40]

Figure 10 shows how a customer can chat with Eno. The chatbot can answer questions about one's account, balances and credit cards and can send notifications, insights or summaries [40].

4. Methodology

As the team of this work consists of two members, the work uses the approach of having one member being familiar with the studied TV series, taking the position of an expert, while the second member stays unknowing of the exact plot to have an unbiased view on the data and results.

The planned procedure is based on the Cross-Industry Standard Process for Data Mining, short CRISP-DM. This process defines work steps that must be performed in a Data Science project. The CRISP-DM consists of the six phases, Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation and Deployment [41], that follow the procedure as it is shown in Figure 11:

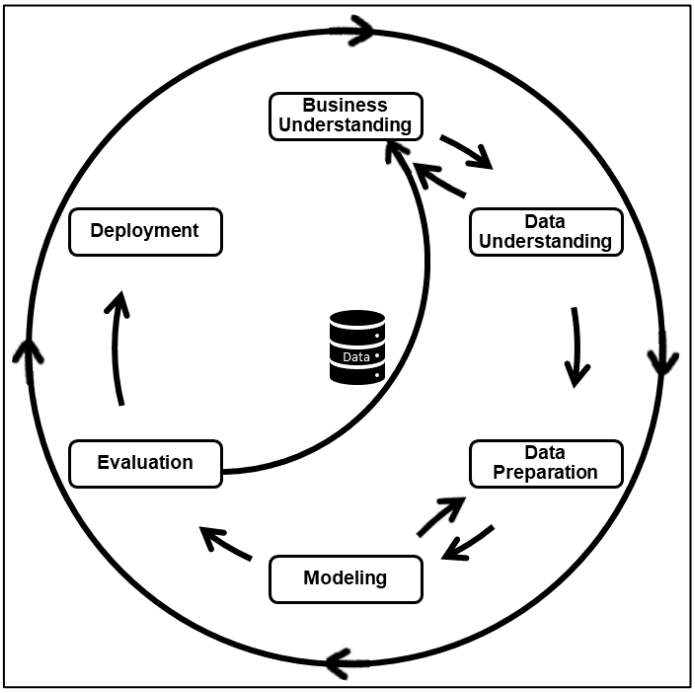


Figure 11: Phases of CRISP-DM [42]

The structure of the paper is based on the CRISP-DM model.

Since this paper does not cover the deployment of a finished product, the last phase, deployment of the CRISP-DM, will be replaced by an outlook. The following sections describe each of the above steps in detail.

4.1 Formulate Hypotheses

This step is based on phase 1 of the CRISP-DM, the business understanding, and is the basis for the project. This phase focuses on understanding the objectives and formulates the hypotheses and/or problem definition [42]. The formulated hypotheses that will be explored in this paper have already been listed in chapter 2, “Problem Statement”, and will be realized in the modelling step (chapter 5.3 "Modelling").

4.2 Acquiring Data

Acquiring the initial data belongs to the data understanding, the second phase of the CRISP-DM.

This phase starts with the collecting of the initial data and familiarizing oneself with it [42].

In the step "acquiring data", the analyst needs to acquire the data and load and integrate it if necessary. For future replications of the project, it is important that encountered problems and the corresponding solutions are documented. Possible problems could be a long lag time or the collection from different sources [42].

The data used in this paper is acquired from the website https://imsdb.com/. This site contains scripts of movies and series in HTML format that are openly available so that the data acquisition is implemented via web scraping to use in a scientific context [43].

The data is stored in a suitable format like a Panadas DataFrame. For the data extraction, the script maps the names and spoken texts to the characters.

4.3 Explore Data

Data exploration also falls under the second phase. This step aims for a better understanding of the data, using querying, visualization and reporting to get a general idea of the data. During the exploration, potential errors like empty cells or content errors can be detected [42].

Metrics such as the average number of words or sentences per person, or the most frequently used words, can be used in this paper. Visualizations such as word clouds or bar charts help for spotting.

4.4 Prepare Data

Data Preparation is the third phase of the CRISP-DM process. This phase handles the construction of the final data set or the data that is used to train the model from the raw data.

In this step, the data is cleaned and prepared for analysis. To achieve good and satisfying results, it is important to use cleaned data, as otherwise, the results are questionable. During the cleaning process, a clean subset can be generated, or estimated missing data can be incorporated. For example, incorrectly formatted, redundant or irrelevant data is removed [42].

Concrete preparation steps for text editing are tokenization, the breaking down of words into smaller parts, and the removal of stop words and frequently occurring words that are irrelevant for text analysis, for example, "the" or "a". Furthermore, unnecessary text elements such as punctuation marks, HTML tags or unknown letters are removed.

4.5 Modelling

The modelling phase is the fourth of the CRISP-DM process. In this step, the processed data is used and modelled to answer the hypotheses from step one. Various modelling technics can be selected and applied to find the approach that works best for processing the data and assessing the hypotheses. Since some technics used for the modelling need specific prerequisites, it may be necessary to go back to the data preparation phase to bring the data into the required format [42].

An example of a model is the Bag of Words (BoW) model. It is used to convert text into numerical values. The model is then used for analyses such as Sentiment Analysis or Topic Generation [44].

This step often includes machine learning approaches for tasks like topic generation. As we do not have vast amounts of data, this paper will use machine learning algorithms that require little data to perform well.

4.6 Evaluation

Evaluation is the penultimate phase of the CRISP-DM. In this last sub-step that will be regarded in this project, the results from the modelling step are evaluated. Here, the quality of the model is recorded, and the hypotheses are answered. To be certain that the model achieves the business objectives – in this paper, that means answering the hypotheses – it is important to thoroughly evaluate and review the model. For example, it has to be made sure that all aspects have been considered and no important point is missing [42].

The steps of the Data Science process are not strictly linear. If, for example, it is determined in the evaluation that the quality of the model is insufficient, the model must be revised and is then evaluated again.

5. Practical Implementation

This section comprises the main part of this paper. It describes the process of answering the hypotheses, from the data acquisition to the data modelling. The basis is the earlier described methodology.

5.1 Data Acquisition

As mentioned in chapter 4.2 “Acquiring Data”, the data is acquired from the Internet Movie Script Data Base (IMSDB). The IMSDB contains several thousand scripts, movies, and television series. The user community updates the scripts. Some movie scripts are just drafts, and others are official scripts [43].

The data is attained by web-scraping imsdb.com. The data acquisition process consisted of three steps: understanding the structure of the webpage, attaining the links to the scripts, and parsing the HTML code of the scripts into Pandas DataFrames.

5.1.1 Structure of the Data

Figure 12 shows how the scripts are accessed on the IMSDB website. The web page contains an overview of all available scripts for the television series Stargate. The episodes are sorted by season. To access a script, first, the overview page of the script is accessed, then the script's HTML. This process is modelled programmatically in python.

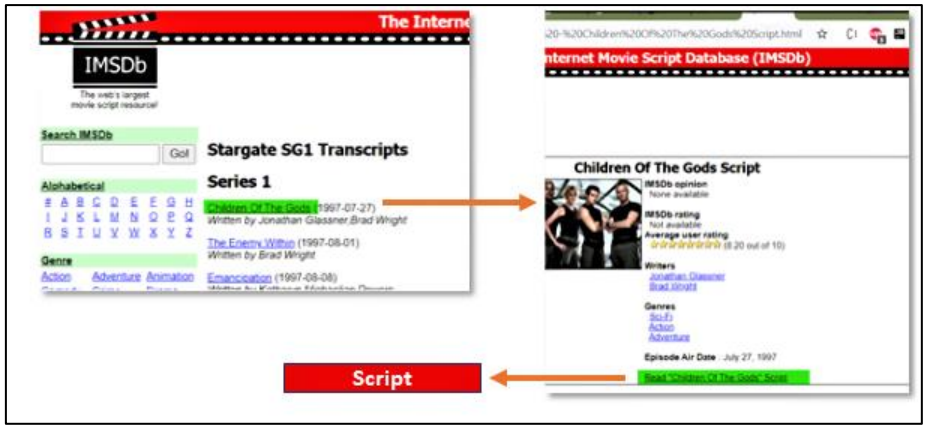


Figure 12: Website structure

The scripts of the Stargate series all have a similar structure. Examining several scripts showed that a character's text is usually preceded by the character's name in HTML-<b>-tags. Their name is preceded by empty <b>-tags. Figure 13 illustrates this structure. The observations about the script are later used to differentiate the character dialogue from the screenplay directions.

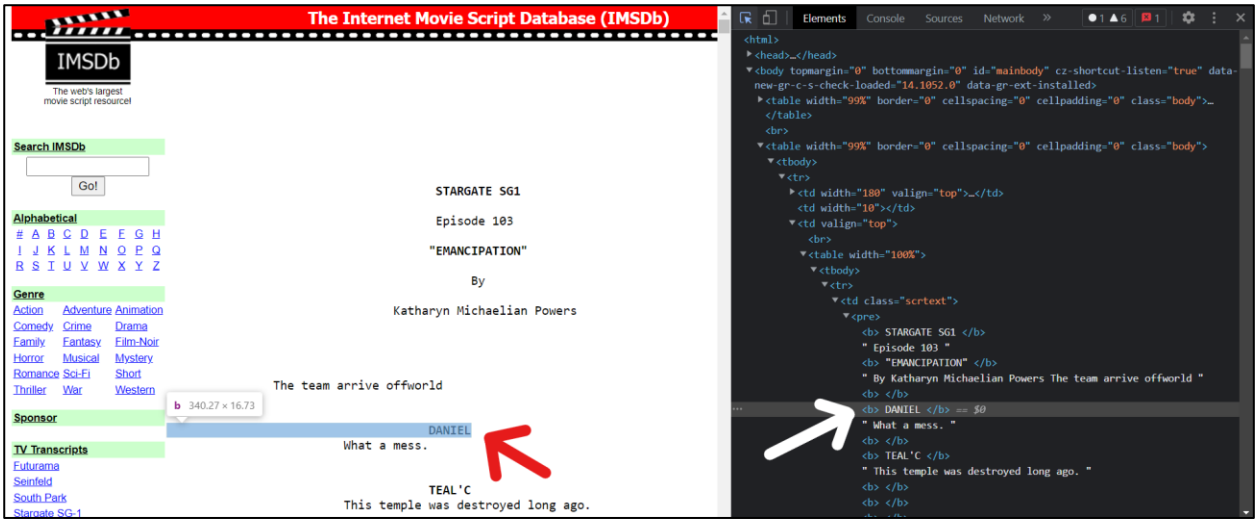


Figure 13: HTML Page IMSDB Script

Listing 1: HTML Code IMSDB Script

5.1.1 Attaining the Script Links

The first step in attaining the data is getting the links to the movie script of each Stargate episode. The python library “beautifulsoup” is used to reach this goal. The HTML "<a>" tags are used to identify the links on the page. On the script overview page, the keyword "Read …" is used to get the link to the actual script.

5.1.2 Parsing the Scripts Code

From the script's HTML code, the character names and texts are extracted and parsed into a pandas DataFrame. The DataFrame has two columns. The first column, "character," contains the name of the current character speaking. The second column, "text", contains one block of dialogue of that character.

5.2 Data Preparation

This section is concerned with preparing the data for the data modelling step. The data preparation consists of two parts: the Exploratory Data Analysis (EDA) and the data cleaning. The following describes the process of both steps.

5.2.1 Exploratory Data Analysis

Peeking Into the Data

After acquiring the data, it is examined. The data shows several inconsistencies. Figure 14 shows examples of the inconsistencies. For instance, certain characters are sometimes referred to with different names. “Samantha Carter” is “Carter”, “Sam”, and “Samantha” (marked in red). Side characters are sometimes referred to as "WOMAN" or "FEMALE". Further, some apostrophes are formatted incorrectly (marked in green). Several text passages are empty, and screen directions are classified as characters (marked in blue).

Same character different name

	character		character
0	interlude	0	JACK
1	FEMALE TECHNICIAN	1	TEAL'C
2	interlude	2	SAM
3	HAMMOND	3	JACK
4	FEMALE TECHNICIAN	4	interlude
5	INT SGC CONTROL ROOM	5	DANIEL
6	interlude	6	interlude
7	O'NEILL	7	JACK
8	HAMMOND	8	interlude
9	TEAL'C	9	JACK
10	DANIEL	10	interlude
11	TEAL'C	11	JACK
12	CARTER	12	interlude
13	HAMMOND	13	DANIEL
14	interlude	14	JAFFA

Faulty format of „ ‘ “-symbol

	text
0	All right kids. We□re...
1	We practically just g...
2	Trees and Moss.
3	Well, a few miles fro...
4	Captain any signs of ...
5	None, sir.
6	Daniel tries to speak and puts ...
7	Ah! Ah! We□ll flag it...
8	O□Neill
9	A man comes out of the tree's a...
10	Help me. They find me...
11	Who?
12	Taldor. They find me ...
13	Daniel?
14	Tal? I don□t, I have ...

Screen directions as characters and empty text fields

	character	text
477	INT SGC GATE ROOM	
478	interlude	[Klaxons are blaring and the St...
479	HAMMOND	Teal'c, you have a go!
480	DREY'AUC	(To the team) I m...
481	DANIEL	Well, you wouldn't have much of...
482	CARTER	But you're welcome in the Land ...
483	O'NEILL	Yep, and they got tons of wide-...
484	interlude	[He hands Rya'c a baseball glove.]
485	RYA'C	W-what is it?
486	interlude	[Rya'c puts the glove on his he...
487	O'NEILL	Well, I guess I'll have to come...
488	interlude	[O'Neill takes the hat off his ...
489	TEAL'C	(To O'Neill) I sh...
490	interlude	[O'Neill nods, and Teal'c and h...
491	THE END	

Figure 14: Data inconsistencies examples

General Statistics

Next, some general statistics are created. The data set has about 72.000 rows. Each row corresponds to one person speaking or an interlude. The serialized data amounts to about 11 Megabytes. There are 1943 unique characters. The top character is “interlude”. Interludes amount to about 20% of the rows. There are about 1300 text passages that are empty.

Word Clouds

These statistics are followed by a word cloud (Figure 15). Very prevalent words are “Teal'c”, “Sam”, “Jack”, and “Daniel”. It is plausible that these are the main characters. Stop words are excluded from the word cloud. However, when removing further seemingly meaningless words like “know”, “look”, and “see”, other words become prevalent. This word cloud is shown in Figure 16. In this cloud, words like “back”, “come”, “now”, “gate”, and “time” imply that characters often speak with urgency about moving to different places. Figure 17 shows that stays true when also removing prevalent character names from the word cloud.

Figure 15: Word cloud all words no stop words

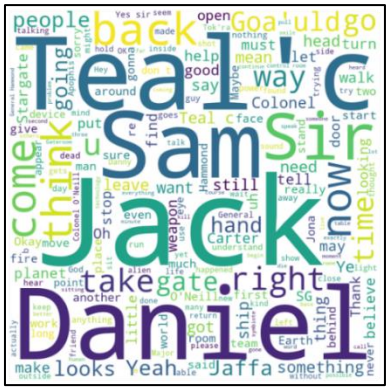


Figure 16: Word cloud all words additional words removed

Figure 17: Word cloud all words, without the most prominent characters

Character Occurrences

Because of the high number of characters, the character occurrences are further inspected. Here Jack (or O'Neill), Daniel, Sam, Carter, and Teal'C are the most occurring characters. This observation is in line with the most prevalent words in the word cloud. The characters that speak the most are also the ones that are spoken of the most. This strengthens the hypothesis that these characters are the main characters. We also observed that the character names are preceded by several white spaces.

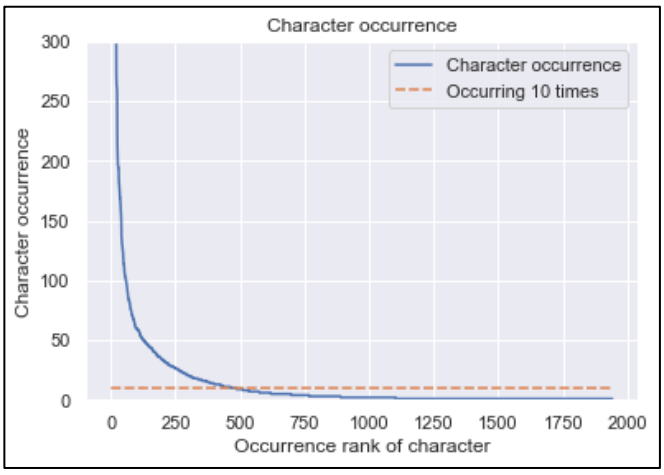


Figure 18 shows that most of the 1943 characters are not occurring often. When sorting the characters by occurrence, there is a steep exponential drop from several thousand to 390 occurrences in the first 20 characters. From the 500th most often occurring character on, characters less than eleven times over the entire series.

Figure 18: Ranked character occurrence

Word Counts

Next, the word counts of the characters are inspected. One apparent observation is that the characters on the tail end of this statistic are primarily comments about the scene. Examples include "INSIDE THE AIRPORT" or "INSIDE THE SHIP". They usually have a word count of 0.

Another observation is that the order of the characters sorted by word count differs from the order of the characters sorted by occurrence. For instance, even though "JACK" appears more often than "DANIEL", "DANIEL" speaks more words.

When inspecting the total word counts of the seasons and episodes, season nine is an outlier. Seasons one to eight have between 20 and 22 Episodes. Season nine has ten episodes. Consequently, season nine has a lower total word count than the other seasons. Figure 19 depicts this difference. However, as shown in Figure 20, the episodes in season nine, on average, have more words than the episodes in the other seasons.

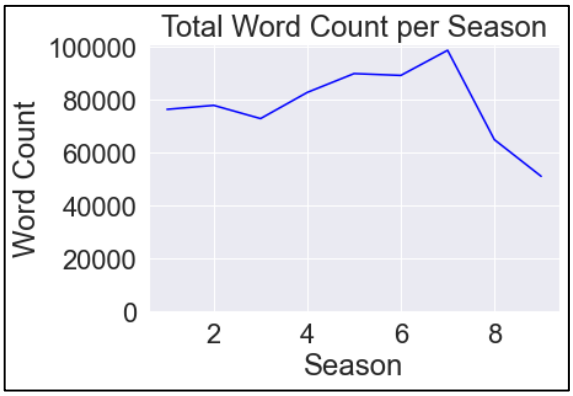


Figure 19: Total word counts over the seasons

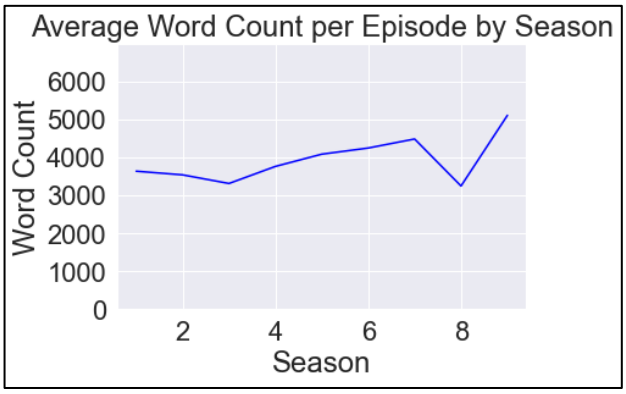


Figure 20: Average word count per episode by season

5.2.2 Data Cleaning

The cleaning process is guided by observations made in the Exploratory Data Analysis. The exact steps are:

- Converting text and character names to lowercase
- Removing access to white spaces
- Removing left over HTML-tags
- Removing punctuation

- Renaming characters

Additionally, all records where the character column contains names that are not actual characters are removed. These names are defined as all names consisting of three or more words, names that occur only once, names where the word count in the text column is zero and other names containing words like “ship”, “the end”, or “scene”. Most of these names are screen directions rather than characters. Removing these records results in a loss of 3% of all records. However, more than half the characters are lost. The number of characters is now 808.

Lastly, all records are combined in one large DataFrame. In order to filter for specific seasons and episodes, the columns “episode” and “season” are added to the DataFrame.

More complex operations such as lemmatization are omitted for now. Each hypothesis may need different transformation steps.

5.3 Modelling

In this subchapter, the hypotheses formulated in chapter 2, "Problem Statement", are assessed, and their results are discussed.

5.3.1 Hypothesis 1 – Determination of Main Characters

The first hypothesis that is assessed is the assumption that one can tell if a character is the main character by the amount of speech they have, meaning that the characters with the highest word count are main characters. The characters are regarded per season since it can be assumed that main characters change throughout seasons by, for example, an actor or actress leaving the cast.

To assess this hypothesis, a function that has the season and its number of episodes as input parameters is developed to propose the main characters and generate a table that shows the assumed main characters.

To achieve this, the function first concatenates all characters and their text from the single episodes of the season. The function then generates an additional column "word_count" to count the words spoken by a character, and groups the table by character, summing up the total number of words per character.

The important part is now to decide, depending on the word count, which characters could be classified as a main character. The list is sorted by the word count in descending order. It is now to decide from which position downwards the characters are considered side characters.

To decide this, there are four different approaches:

- The border between the last main character and the first side character can be detected through the delta between their word counts being at least 60% of the word count of the last main character. The value of 60% was determined through testing.
- Considering that a typical series has four or five main characters [45], the five characters with the highest word count are considered main characters.
- Each character that has a word count of at least 7000 words in a season is a main character. The value got determined through the data exploration.
- The final approach combines the first two approaches. The main characters are determined by the 60%-rule, but there are at most five of them.

To identify the best approach, the main characters of every season are proposed with the function and displayed in Table 1. The table only shows the first three seasons as an excerpt.

	season	approach	Assumed main characters	Number of main characters proposed
0	1	60% rule:	[jack_o_neill, daniel_jackson, samantha_carter]	3
1	1	top 5 are main characters	[jack_o_neill, daniel_jackson, samantha_carter, hammond, teal_c]	5
2	1	everyone with >=7000 words is a main character	[jack_o_neill, daniel_jackson, samantha_carter]	3
3	1	combined 1+2	[jack_o_neill, daniel_jackson, samantha_carter]	3
4	2	60% rule:	[samantha_carter, jack_o_neill, daniel_jackson, hammond, teal_c]	5
5	2	top 5 are main characters	[samantha_carter, jack_o_neill, daniel_jackson, hammond, teal_c]	5
6	2	everyone with >=7000 words is a main character	[samantha_carter, jack_o_neill, daniel_jackson, hammond]	4
7	2	combined 1+2	[samantha_carter, jack_o_neill, daniel_jackson, hammond, teal_c]	5
8	3	60% rule:	[jack_o_neill, daniel_jackson, samantha_carter]	3
9	3	top 5 are main characters	[jack_o_neill, daniel_jackson, samantha_carter, teal_c, hammond]	5
10	3	everyone with >=7000 words is a main character	[jack_o_neill, daniel_jackson, samantha_carter]	3

Table 1: Main characters that are proposed per season per approach

Different approaches return varying numbers of characters. The following Figure 21 shows the distribution of the values.

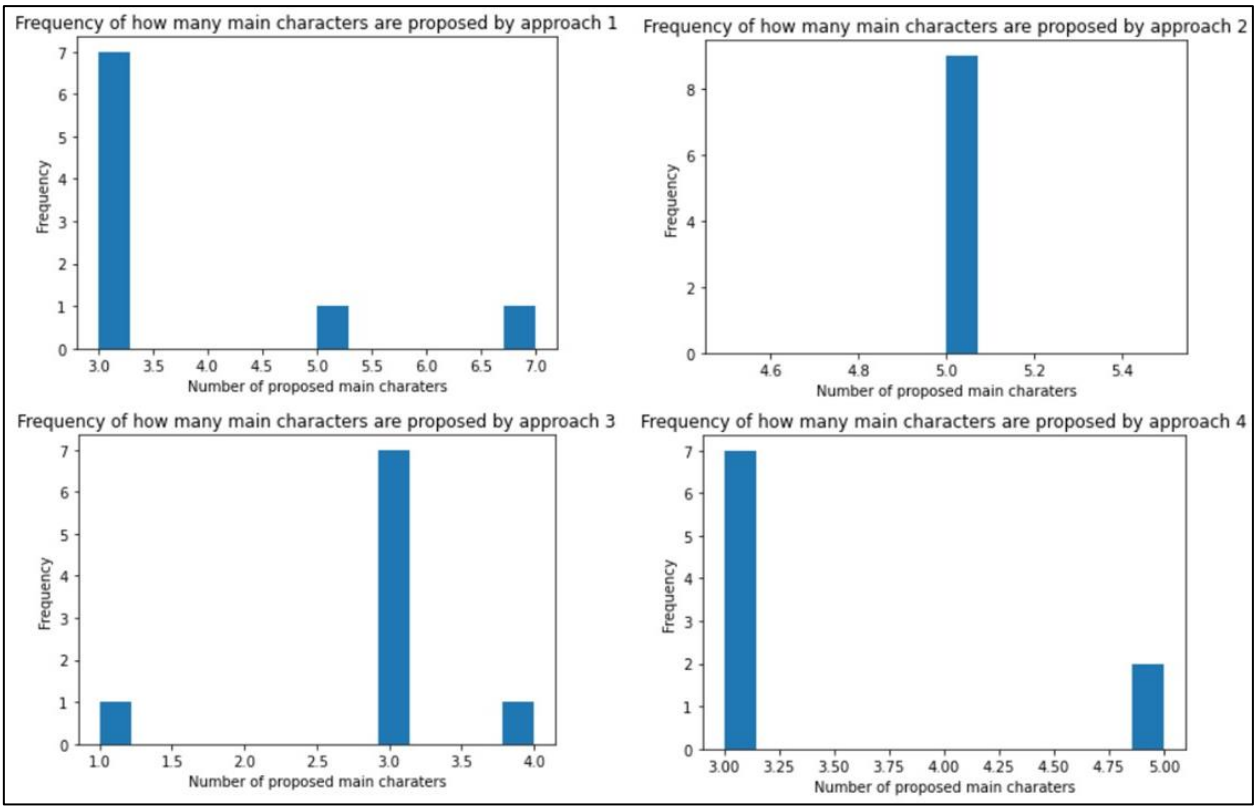


Figure 21: Number of characters proposed per approach

As seen in Figure 21, the first approach detects between three and seven main characters per season. However, this approach is at risk of proposing a lot more main characters should no gap between two characters be bigger than 60% of the first character's word count.

The second approach always returns five main characters, while the third approach ranges from one to four characters.

The last approach that combines the 60%-rule and limitation to five main characters returns between three and five characters. This approach also solves the disadvantage of the approach that uses only the 60%-rule. If there is no gap found that fits the criteria, this approach automatically limits the main characters to five, thus preventing a main character count of for example 100.

Overall approaches, the algorithm finds, on average 3.75 main characters in a season. To verify the accuracy of the algorithm, it is also important to examine the names of the main characters and compare them to the expert's knowledge about the main characters. One fact that stands

out is that independent of the approach used to find the main characters, the algorithm fails in ca. 61% of the cases to recognize the character "Teal'C" as a main character throughout all seasons. The special thing about "Teal'C" is that the character is mostly non-verbal. He communicates primarily through facial expressions and speaks up rarely [46].

Table 2 below shows how often the "Teal'C" is recognized as a main character.

Approach	Number of times how often Teal'C is recognized as main character
Approach 1 – 60%-rule	2 / 9
Approach 2 – the first 5 characters	8 / 9
Approach 3 – all characters with >= 7000 words spoken	0 / 9
Approach 4 – 60%-rule with max. 5 characters	1 / 9

Table 2: Amount how often "Teal'C" is labelled as main character per approach

From this perspective, approach two, which always assumes the top five characters to be the main characters, performs the best.

To summarize this, the algorithm with the different approaches shows that it is only possible to recognize a character as a main character when they speak often. However, when a character communicates through facieal expressions and body language, such as for example "Teal'C" [46], it is not possible.

Therefore, the hypothesis that one can tell if a character is a main character by the amount of speech they have, has to be rejected.

5.3.2 Hypothesis 2 – Topics are Mainly Science Fiction

Stargate is a science-fiction series [47]. In accordance with this, the second hypothesis states that most of the topics discussed are about science-fiction. The following sections describe the process of examining this hypothesis. The hypothesis is examined in two aspects. First, the number of topics in the field of science fiction is evaluated. Second, the most prevalent topic is discovered. The following sections describe the topic modelling process and the resulting topics. Lastly, the topic model is applied to the script data to determine the most prevalent topic.

Topic Modelling Process

The topic modelling process is composed of three steps. First, a topic modelling technique is chosen. Second, the data is transformed into the correct format. Third, the model is created and the hyper-parameters optimized.

As described in section 3.3, "Topic Modelling", there are several methods for topic modelling. Non-Negative matrix multiplication could potentially be suited, as it is suited for shorter texts. However, the length of the scripts potentially succeeds the length of texts that Non-Negative Matrix Multiplication is commonly used for (headlines, search results, tweets). Latent Dirichlet Allocation is better suited for longer texts. Further, it is the most popular and highly studied topic model. There are many libraries that facilitate creating, evaluating, and improving the models. As also described in section 3.3, LDA addresses many shortcomings of other topic modelling techniques. For these reasons, the topics of this project are modelled using Latent Dirichlet Allocation.

The input of a topic modelling process are documents. The challenge for the topic modelling process is defining the scope of a document. A document could be the text from each season, each episode, each scene or each line of a character. As described in section 3.3 "Topic Modelling", documents are a collection of words that revolve around a set of topics. The question at hand is, at what granularity are the topics most atomic and the texts not too short for the modelling algorithm? The answer to the question is assumed to be a document scope per scene. The differences per scene are considered to be greater than between episodes. Scenes have different

characters, places and hence likely different topics. This topic's automaticity diminishes when regarding episodes or seasons. Furter a scene contains more text than a single line of a character, which will be too short for the LDA algorithm to work properly. Figure 22 compares the coherence scores of the Latent Dirichlet Allocation with documents on the episode level and documents on the scene level. The graphic shows that the coherence scores of the former approach are mostly lower than the latter. This is especially true for lower topic count numbers.

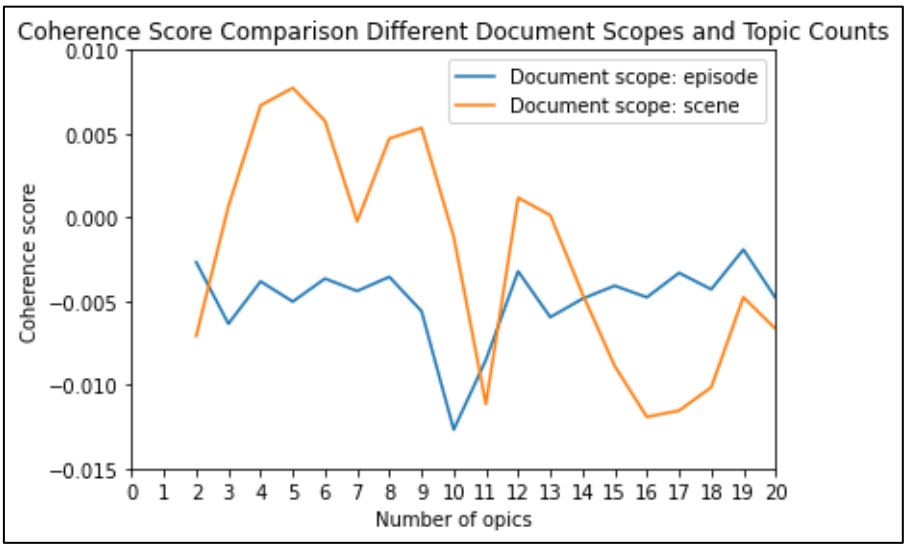


Figure 22: Comparing coherence scores of LDA topic models with different document scopes

For the above-outlined reasons, a document scope per scene is chosen. A scene is defined as the text between two interludes. This approach is not perfectly accurate, as interludes do not strictly separate scenes. However, there is usually at least one interlude between scenes, which makes the granularity larger than the character line document scope.

At this point, it must be noted that the document scope was first defined per episode. For this, the LDA approach is suited as episodes contain rather much text. The switch in the document scope came about later in the project. Here, a different topic model could have been considered, as the document scope now contains lesser amounts of text. Non-Negative Matrix Factorization (NMF) could be an option, as it is suited for shorter texts. Figure 23 compares the coherence scores of Latent Dirichlet Allocation topic models with different topic counts with the corresponding Non-Negative Matrix Factorization models. The figure shows that NMF seemingly performs better than LDA. This is especially true for higher topic counts. However, whether there

is an optimized NMF model that performs better than the optimised LDA model used for the project needs further investigation.

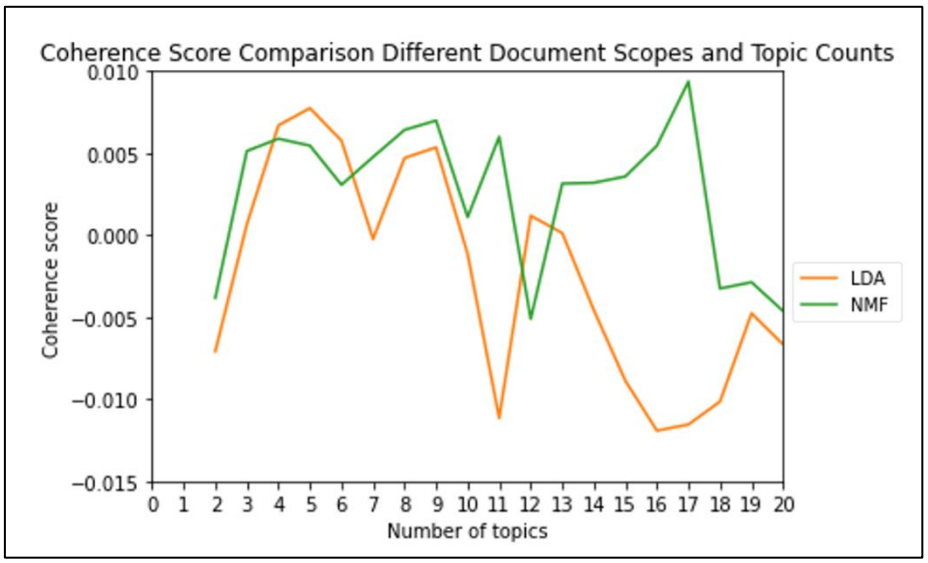


Figure 23: Comparing coherence scores of LDA and NMF topic models

After creating the documents and performing additional data cleaning steps like removing stop words, stemming and lemmatization, the Latent Dirichlet Allocation model is generated and adjusted. The hyperparameters to tune are the number of topics, the number of iterations for the training, and the number of irrelevant words to remove from the documents. The hyperparameters are adjusted using a lesser extensive version of grid search. Instead of testing all possible variations of parameters, as described in section 3.3.4 “Improving Topic Models”, the parameters are adjusted one after another. This is done to one save time and computing resources. The resulting topic model may not be the best topic model out of all possible options; however, the resulting model will be an improvement over the base model. This suffices for the purposes of this project.

The first parameter to optimise is the number of topics. As mentioned, Figure 23 shows the coherence scores of LDA topic models with topic counts. It shows that when regarding models with topic counts between two and twenty, five is the optimal topic count. Twenty is chosen as the upper limit for the regarded topic counts because this number of models can be created in a reasonable amount of time. Further, the coherence scores generally decrease with higher topic counts. Hence, we regard testig higher topic counts as obsolete.

Table 3 shows the 15 most important words for the selected five-topic LDA model. Inspecting these words shows that several words like “know”, “one”, or “sir” appear in several topics. An overlap in topics makes distinguishing topics harder.

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
1	look	go	daniel	right	im
2	think	know	sir	come	us
3	know	well	well	ship	think
4	well	yeah	know	go	know
5	say	sir	teal	teal	god
6	one	tell	im	colonel	take
7	take	need	see	time	kill
8	goal	look	jackson	neil	peopl
9	whi	daniel	wait	one	ffa
10	sir	let	say	sir	goal
11	us	find	back	cater	one
12	good	thank	gate	im	well
13	jack	im	go	ffa	believ
14	time	one	yes	door	understand
15	see	take	somet	know	make

Table 3: 15 Most significant words for a LDA topic model

In the context of topic modelling, words that appear in many documents are similar to stop words. These words appear often but do not contribute to uniquely defining a topic. With this, we define a topic as better if the words inside the topic are unique to the topic. Therefore, words that appear in many documents are filtered out in the next step. However, filtering out the words that generally appear most often is not desirable. A word may often appear in a single document but not in any other document.

For this reason, we use the Tf-idf described in section 3.3.2 “Topic Modelling Techniques” to filter out all words that carry little weight for defining a document and, hence, a topic. Twenty topic models with five topics are examined to define the best amount of words to remove. The models are created by removing words with the zero to 20% lowest Tf-idf scores from the documents. Figure 26 shows that the coherence score diminishes when removing more words with low Tf-idf scores. This makes sense, as the coherence is based on words that often appear together, and such words have low Tf-idf scores.

Because we aim to improve the model by making the topics more distinguishable, we use a different metric to tune the hyperparameter “percentage of low-value words to remove”. Figure 24 and Figure 25 show part of the LDAvis visualizations of two LDA models with five topics. Both graphics show the visualization of the 30 most relevant words of each topic. Figure 24 displays the topics of a model with no words removed that have a low Tf-idf score. Figure 25 shows the topics of a model with 8% of the lowest Tf-idf scoring words.

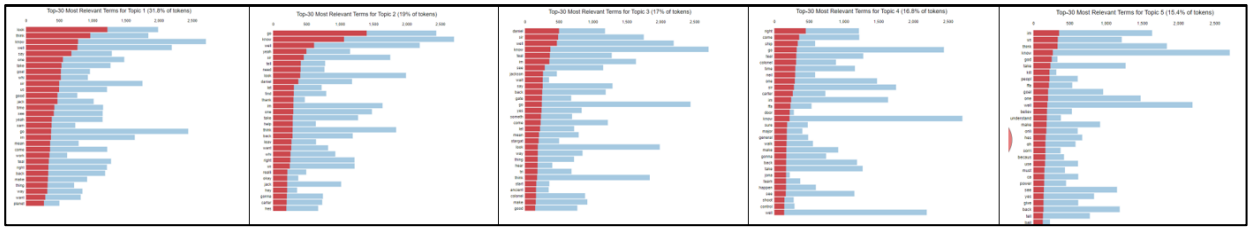


Figure 24 LDAvis of vísualization five-topic LDA model no Tf-idf threshold

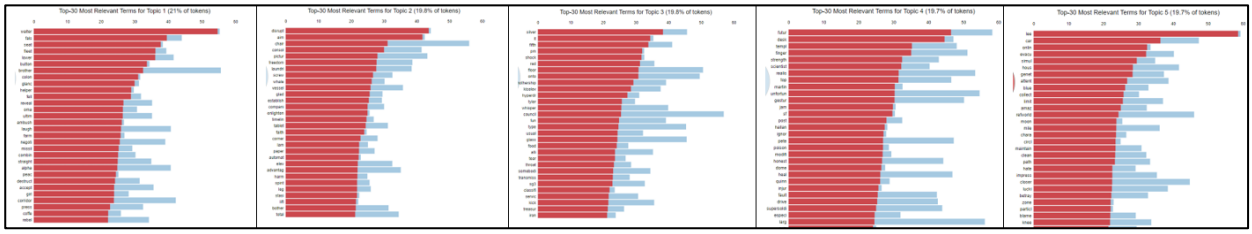


Figure 25: LDAvis of vísualization five-topic LDA model eight percent Tf-idf threshold

The figures illustrate that removing words with a low Tf-idf score results in words mostly appearing in lesser topics. This is represented by the proportion of the red bar within a blue bar. As described in section 3.3.3 “Evaluating Topic Models”, the blue bars represent the total frequency of a word. The red bars represent the estimated frequency of a word within a given topic. A high proportion of red bars within the blue bars indicates that the words do not overlap much across topics. Thus, the topics are more unique. As this is what we want to mesure, we use the mean proportion of red to blue bars, or “term frequency proportion per word in a topic”, as a metric to tune the Tf-idf threshold hyperparameter. Figure 27 shows this proportion for different Tf-idf thresholds. This proportion increases as more low Tf-idf-scoring words are removed. The curve flattens at around 85%. Hence, we choose the model with the first Tf-idf -threshold that yields the just described proportion of above 85%. This is the case when removing 8% of the words with the lowest Tf-idf score.

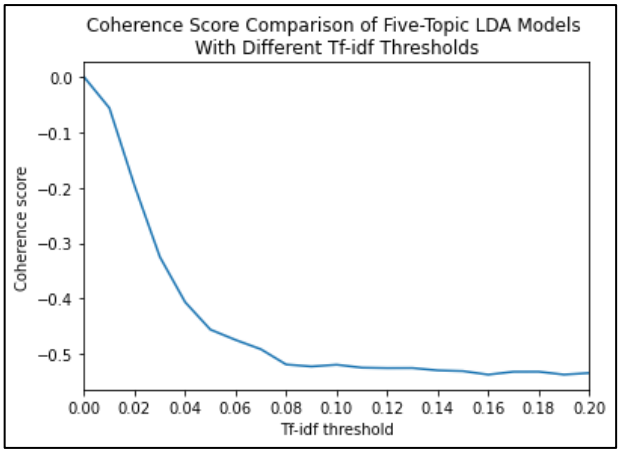


Figure 26: Coherence score comparison of LDA models

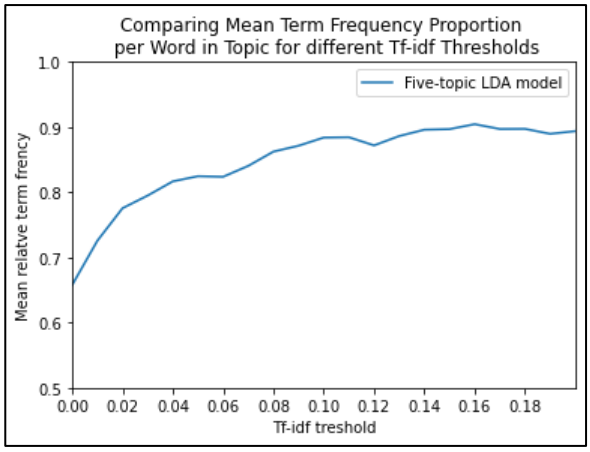


Figure 27: Comparing term frequency proportion per word in topic

As for the hyperparameter of the number of iterations the model is trained, Figure 28 shows that the coherence score generally increases as the number of iterations increases. However, five iterations yield a relatively high coherence score. The next best value is ten passes. Because of our limited computing capabilities, we decide to train the model on five passes.

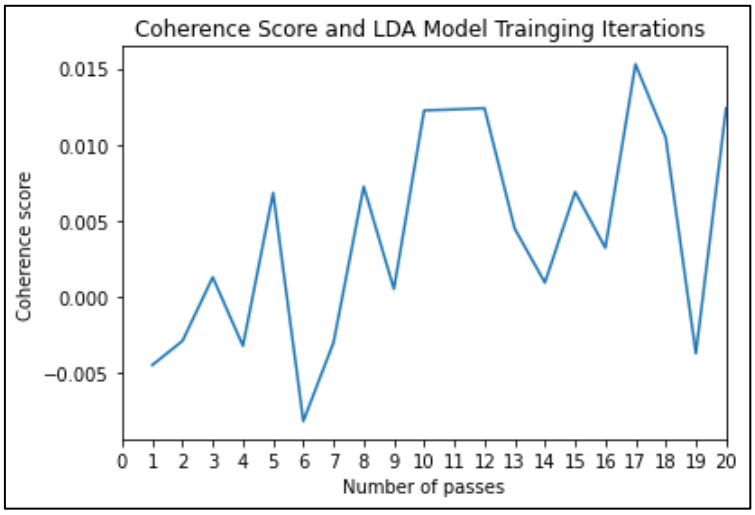


Figure 28: Coherence score and LDA model training iterations

Interpreting the Resulting Model

Interpreting and labelling the topics of the topic model needs human evaluation. For this, both researchers, the expert and the blind researcher label the topics and compare them. The results are shown in Attachment B “LDA Topic Model Labelling and Comparison”. Both interpretations are fairly similar. The topics are labelled: “Negotiation”, “War”, “Faith”, “Sci-fi and injury” and

“Fighting”. For comparison, the topics of models with lower Tf-idf scores are also labelled. One observation is that finding topics is easier with fewer low-importance words removed. However, the interpretations have a bigger variance. Finding labels when more low-value words are removed is harder. However, the labels of both researchers are more similar. This is presumably because removing “filler” words results in less coherence and makes finding a label harder. However, the topics are more specific.

The first aspect of the regarded hypothesis states that most topics are in the field of science-Considering that only one of the five topics in the selected mode is related to science-fiction, the first aspect of the hypothesis is rejected. The topics revolve more around war, sneaking around and travelling. The expert researcher confirms this observation. The series revolves much around war and the military.

To validate the topic model three random episodes are selected. The expert researcher compares the assigned topic distribution with the script content.

Season 3 Episode 19

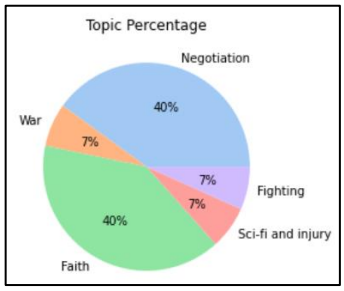


Figure 29: Topic distribution season 3 episode 19

Season 6 Episode 7

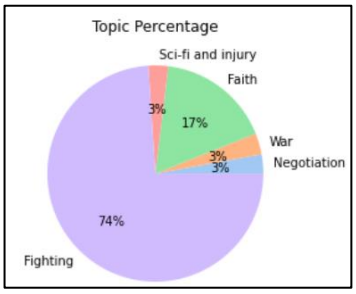


Figure 30: Topic distribution season 6 episode 7

Season 7 Episode 15

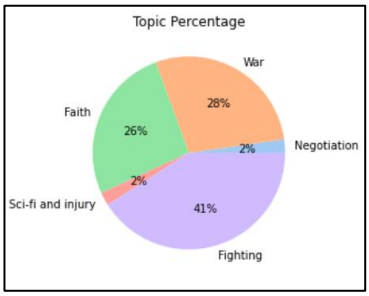


Figure 31: Topic distribution season 7 episode 15

When applying the model to the first episode examined, season three episode 19, the model shows the topics negotiation and faith as the most prominent topics (Figure 29). On expectation this topic distribution is senseful. The episode revolves about an alien race that is regarded as a deity by many in a certain place. Further the team is captured. Because the team tries to free themselves, this circumstance explains the topic prevalence of “Negotiation”.

For the second episode examined, season six episode seven, the most prevalent topic is fighting (Figure 30). However, the episode revolves more around war and negotiation. When examining the topic of fighting again, it shows, that it contains words that indicate precious elements such

as silver, glass, treasure, and iron. Regarding this the topic does fit as the episode is about receiving a rare kind of fictional metal in exchange for knowledge to win the war. The topic labelling process for this topic, however, is the hardest out of all topics. Considering this evaluation, the topic may be renamed “Fighting, valuable goods”. Nonetheless, this point shows the core problem of topic modelling, already described in section 3.3.3 “Evaluating Topic Models” and the paper “Reading Tea Leaves: How Humans Interpret Topic Models” [30]. It is difficult to interpret topic models that are evaluated using automated metrics. The best metrics are based on human judgment. However, using such measures is out of the scope of this project. Further, the current model is an improvement over the base model and still mostly interpretable. Hence, the current topic model is still used, even if it is not perfect.

According to the model, the most prevalent topics in the last examined episode, season seven episode 15 are Fighting (or “Fighting, valuable goods”), “War”, and “Faith” (Figure 31). This topic fits as the plot of the episode includes an alien using a device to manipulate a main character in his sleep. They try to catch them, thus the topic of fighting emerges.

Classifying the Data

The genism LDA model implementation allows applying a given model to a new set of texts. The output is a vector that describes the percentage of each topic of the model in a given text. To classify the texts belonging to a topic, we define the topic with the highest percentage as the main topic in the text. The results for all texts per season are shown in Table 4. It shows that there is a significant change in the seasons and their topic distributions. The same goes for the episodes. For instance, the percentage of the topic “War” ranges between 16% and 25% per season. The second part of the regarded hypothesis states that science-fiction related topics are most often talked about. As the only science-fiction related topic “Sci-fi and injury” is rarely above 25%, the second part of the hypothesis is also rejected.

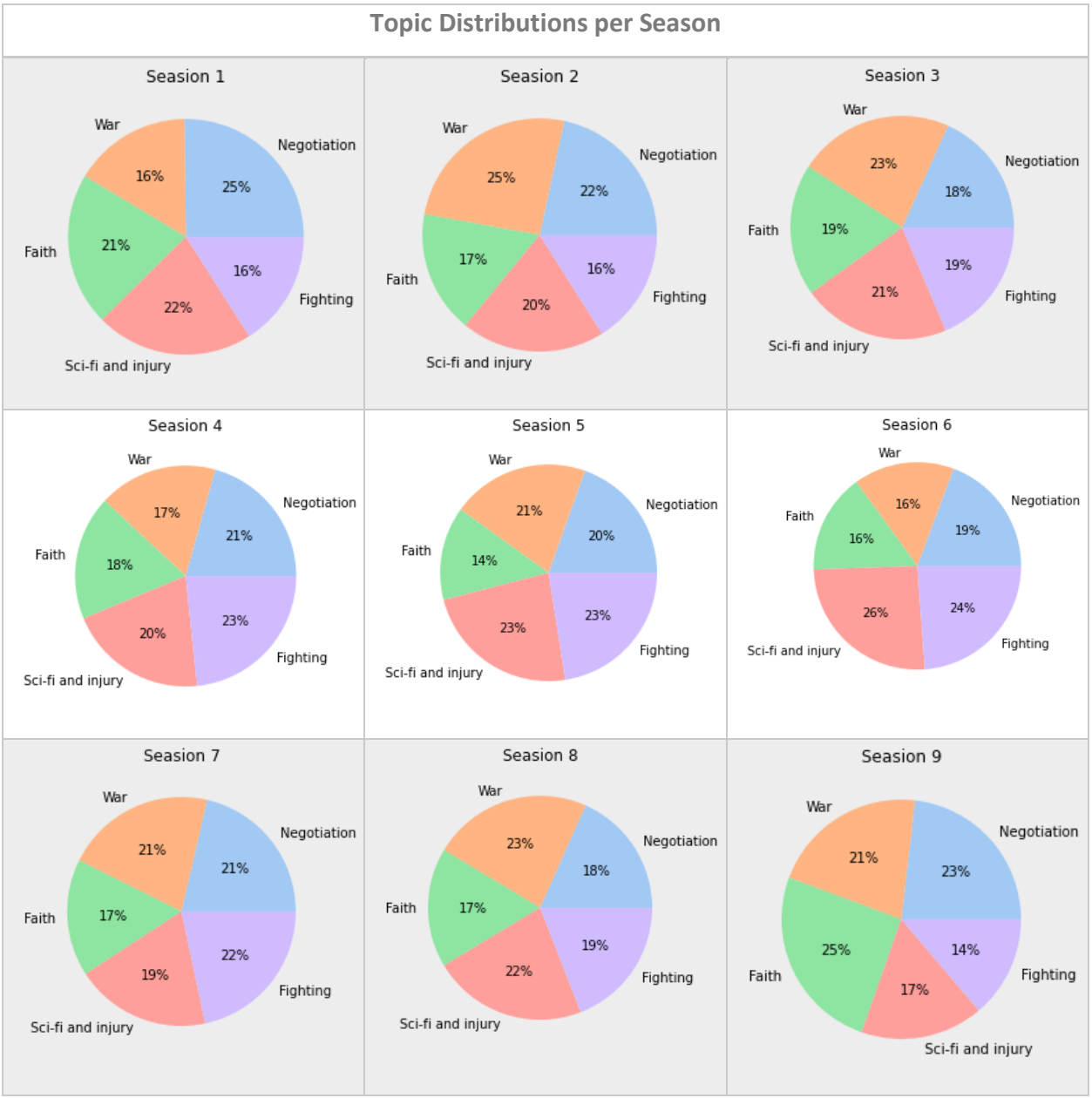


Table 4: Topic distributions per season

5.3.3 Hypothesis 3.1 – Differences in Male and Female Characters: Speech Amount

As mentioned in chapter 2 “Problem Statement”, our second hypothesis is focused at the differences in male and female characters. Examined are differences in the proportion of speech as well as the topics. The following describes the investigation of this hypotheses.

Assigning Genders

To analyse differences in gender, the characters must first be assigned a gender. For this, the python gender guesser library is used. In the data cleaning process, some names were swapped out with longer names containing underscores. Because the gender guesser fails to assign genders to these names, they are split, and only the first name is selected. The gender guesser assigns genders to about 21% of the characters. The rest are labelled as "unknown". The assigned genders are "male", "female", "mostly_male", "mostly_female" and "andy", which means the name is suited for male and female persons [48]. The output is checked manually. Figure 32 shows an excerpt of characters and their assigned gender.

character	gender
samantha_carter	female
jack_o_neill	male
warner	male
daniel_jackson	male
boy	male
warren	male
casey	andy
catherine	female
martha	female
earnest	male
cole	male
cassie	female
davis	male
hanno	male
man	andy
lya	female
harlan	male
walter	male
kennedy	mostly_female

Figure 32: Characters with genders assigned by python gender guesser

To assess the accuracy of the gender guesser, a sample of 38 of the assigned genders is validated by searching the character name in the stargate wiki [49]. Checked are all 18 names assigned "mostly_male" or "mostly female" and a sample of 20 names from the remaining characters. All genders that cannot be validated are counted as falsely assigned. Reasons for a failed validation are that names are not found in the wiki or there is no information on the character's gender.

The gender guesser correctly assigned 40% of the genders in the clearly male or female sample. This percentage was higher in the names classified as mostly male or mostly female. However, there were only 18 names that fell into this category. Hence, this result can be subject to

randomness. When weighting these results according to the number of characters assigned clearly or mostly male and female, the gender guesser fails to assign the correct gender more than 50% of the time. This means it is worse than chance and can therefore not be used for the task of assigning genders to the character names.

Another approach to assigning the genders is searching each character's name in the stargate wiki by adding the name to the Uniform Resource Locator (URL), like in Figure 33 [49]. Then the gender can be extracted from the character description.

Figure 33: Stargate character description page of Samantha Carter [49]

This approach, however, is problematic for several reasons. One is that not all characters can be found by adding the name to the URL because not every character has a description page. Further, some names are assigned to multiple characters. Lastly, some names have spelling mistakes, for example, Antaeus is sometimes called Anteaus. It must be noted that the data for this project is static, meaning that no new data will be added.

For all reasons above, the genders for all characters in this project are assigned manually by searching the stargate wiki. With this approach, 589 characters were assigned clearly female or male. These are about 73% of all identified characters. All others are assigned the gender "unclear" due to the reasons described above. The fact that about one-fourth of the characters have no clear gender is not problematic. This is because they in total, only have a speech proportion (word count) of about 8% over the entire series.

Differences in the Proportion of Speech

After assigning the genders, the proportion of speech between males and females is examined. Over the entire series, male characters speak about 66% of the words and females about 26%. This concludes that male characters have a higher proportion of speech than female characters. This stays true even when hypothetically including the 8% of words spoken by characters of unclear gender. Hence, we accept this element of the first hypothesis. This is demonstrated in Figure 34 and Figure 35.

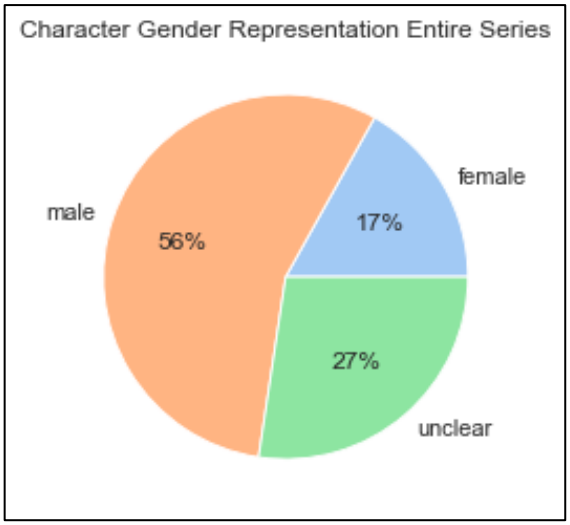


Figure 34: Gender representation over the series

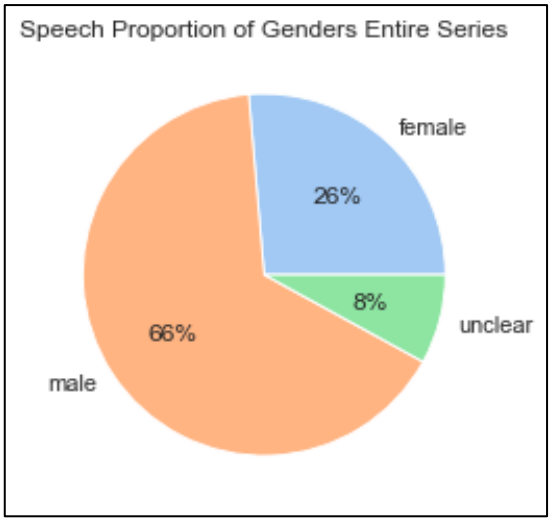


Figure 35: Speech proportion of genders over the series

When regarding the seasons, the total word counts of male and female characters have a similar proportion. Figure 36 shows that male characters speak about twice as much as female characters. In later seasons, however, the speech proportion of characters of unknown gender increases.

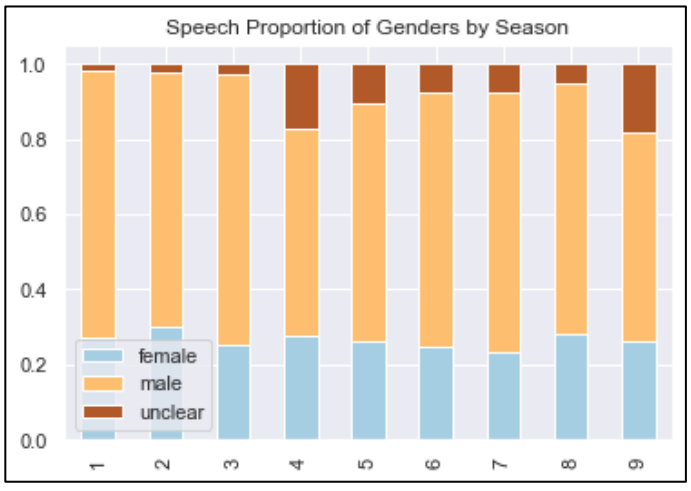


Figure 36: Speech proportion of genders by season

When examining the single episodes, the same observations from before can be made. There are only a few outliers where female characters speak more than 50% of the words. These outliers are marked in Figure 37.

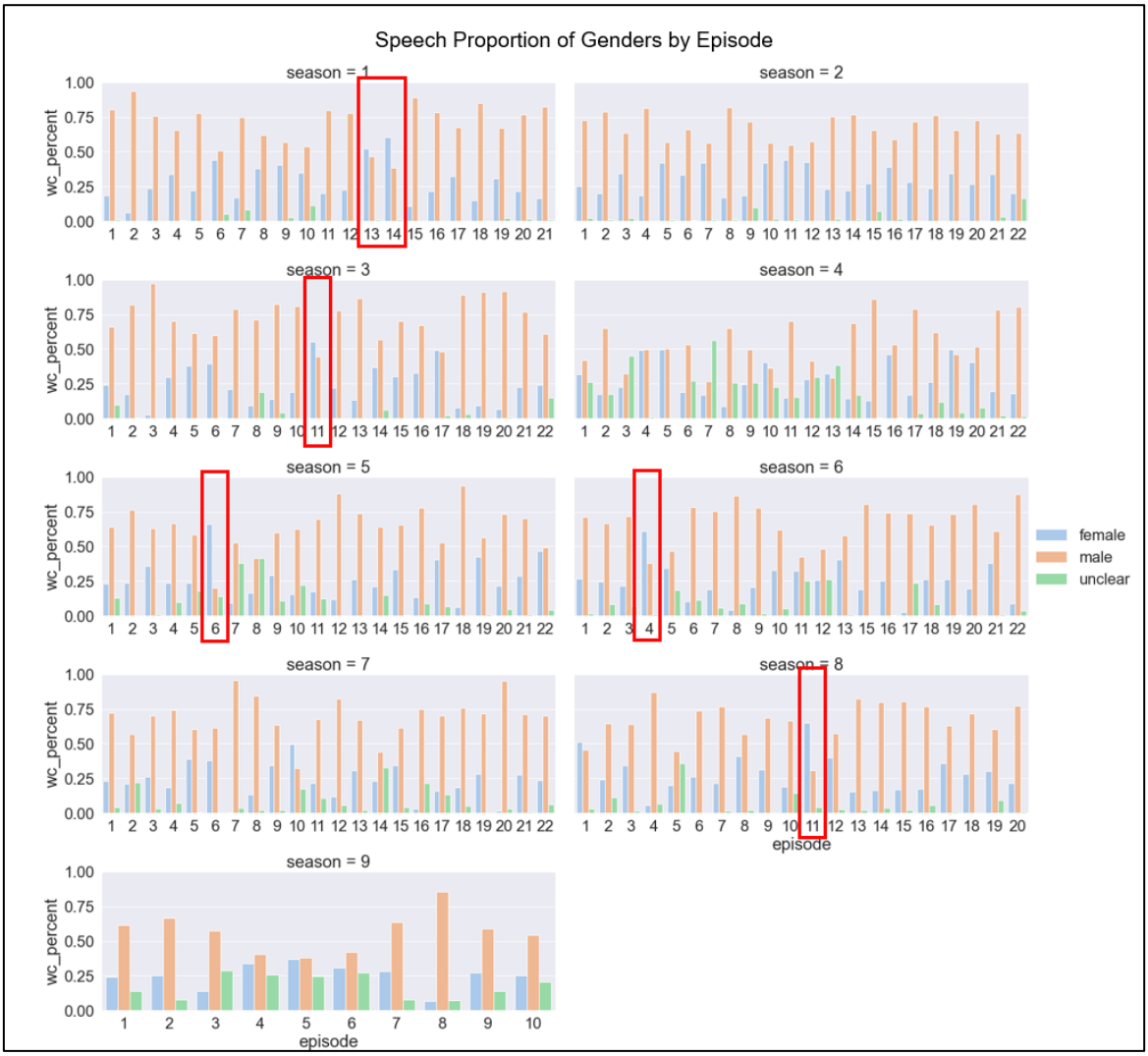


Figure 37: Speech proportion of genders by episode

These outliers are caused by the plot of the episodes. In episode four of season six, for instance, there is a virus outbreak. The main characters of that episode are two female doctors and Samantha Carter. They account for most of the speech content. Episode six of season five revolves around a teenage girl and Samantha Carter. Episode 11 of season eight is about Samantha and her clone. Most of the dialogue is the interaction between the two. The high speech proportion of characters of unknown gender comes from mistakes in the Script. Several characters are not written with their full name but rather abbreviations like “GH” or “TS”. It is not clear what their real name is. Hence, their gender cannot be determined [3].

5.3.4 Hypothesis 3.2 – Differences in Male and Female Characters: Topics

This hypothesis states that men and women talk about different topics. Figure 38 shows the distribution of topics of all spoken words per gender. It shows that female and male characters talk about the same topics in around the same amount. Hence this hypothesis is rejected.

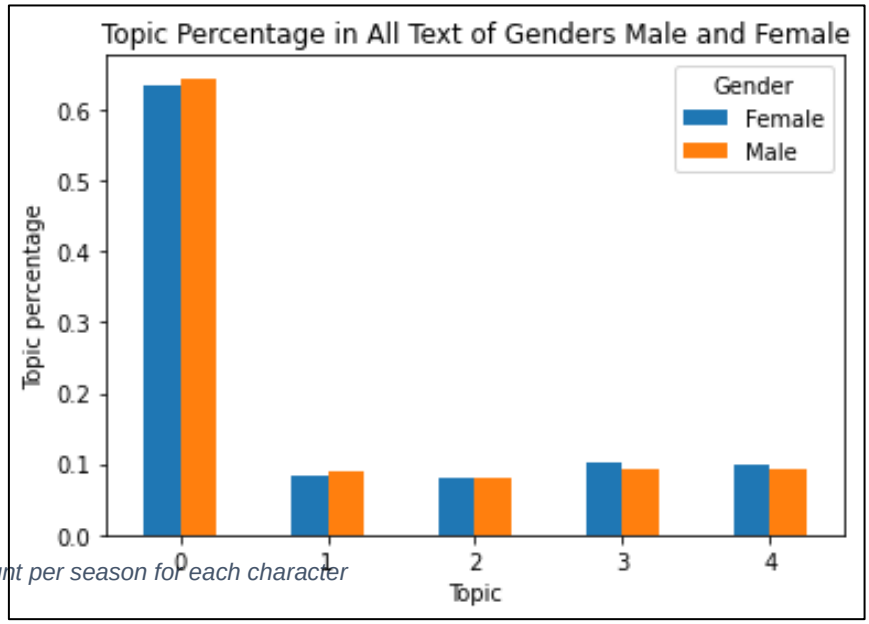


Table 5: Word count per season for each character

Figure 38: Topic percentage in character speech by gender

5.3.5 Hypothesis 4 – Amount of Speech Variation Over the Seasons

This hypothesis assumes that the amount of speech a character has varies significantly over the seasons.To assess this, an algorithm is created that collects all characters over the seasons into one column. This column is then joined with the DataFrames that each contain all characters of one season and the character's word count in this season. These joins use an outer join so that a character that does not appear in all seasons stays in the table. Cells that, thus, do not have a value are then edited to contain the number 0 instead of NaN.

	character	word_count_S1	word_count_S2	word_count_S3	word_count_S4	word_count_S5	word_count_S6	word_count_S7	wor
0	jack_o_neill	14951.0	12524.0	12901.0	14949.0	15294.0	12245.0	9666.0	
1	daniel_jackson	13719.0	11632.0	10949.0	4986.0	6135.0	2629.0	14139.0	
2	samantha_carter	12411.0	14331.0	10939.0	13961.0	14956.0	14744.0	14756.0	
3	hammond	6523.0	7323.0	4145.0	2826.0	5224.0	4834.0	5019.0	
4	teal_c	5631.0	5682.0	4490.0	5058.0	4205.0	4746.0	4308.0	
...	...	...	...	...	...	...	...	...	
806	haikon	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
807	hale	0.0	0.0	0.0	0.0	0.0	246.0	0.0	
808	haley	0.0	0.0	0.0	576.0	317.0	0.0	0.0	
809	hamilton	0.0	0.0	0.0	484.0	0.0	0.0	0.0	
810	zippy	0.0	0.0	502.0	0.0	0.0	0.0	0.0	

811 rows × 10 columns

The resulting table has 811 rows and ten columns, and an extract is shown in Table 5. The rows are sorted in descending order by the word count in season 1.

For better visualization, the first five characters of every season are extracted and displayed in a line chart.

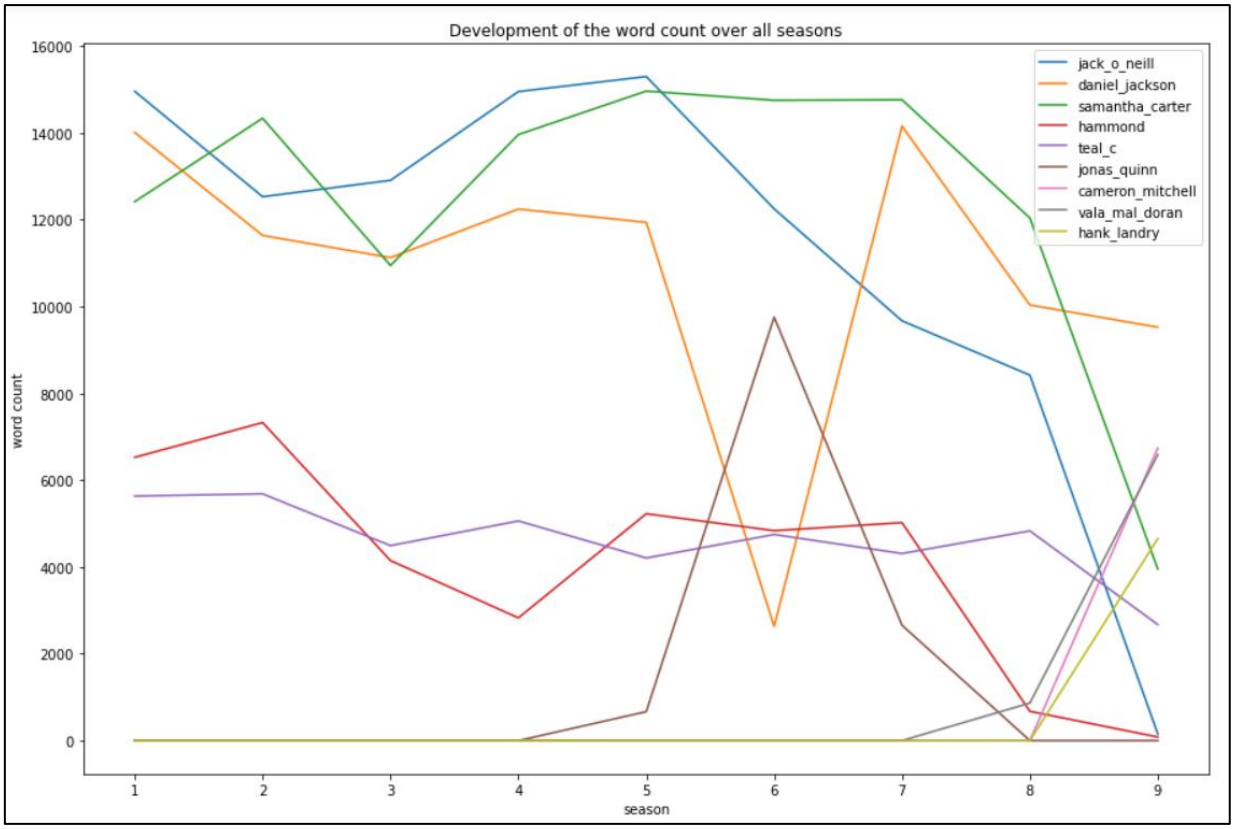


Figure 39: Development of the word count over all seasons

Figure 39 shows how the word count of the first five characters varies over the nine seasons. The x-axis shows the season, while the y-axis shows the word count. Each line belongs to a different character.

One can see that the number of words spoken varies greatly over the seasons. The only character whose word count remains relatively constant is that of “Teal’C”, who, as already mentioned in chapter 5.3.1 “, Hypothesis 1 – Determination of ”, does not rely on words but rather uses facial expression and body language [46].

The biggest change in the word count has the character “Daniel Jackson”. The character’s word count has a sudden drop in season six, where the actor had left the series for one season due to concerns about his character not being allowed to show his full potential [46]. This reason is also mapped in Figure 39, as the word count of Daniel Jackson (orange line) decreases from the first to the fifth season. Looking at the brown line showing the word count of “Jonas Quinn”, which peaks exactly where Daniel Jackson’s word count is at its lowest, allows the assumption that this character has taken over Daniel Jackson’s spot in the series. This assumption is confirmed by the

expert [46]. The word count of Jonas Quinn decreases until 0 in season 8 with the return of Daniel Jackson.

The upper graphic displays the total words spoken by each character in a season. However, since seasons can have a different number of episodes or just a different count of the words spoken in the whole season, to really compare the word count of a character over the seasons, the word count has to be seen relative to the total amount of words spoken.

Setting the word count per character in relation to the total word count, the following line chart, Figure 40, is the result.

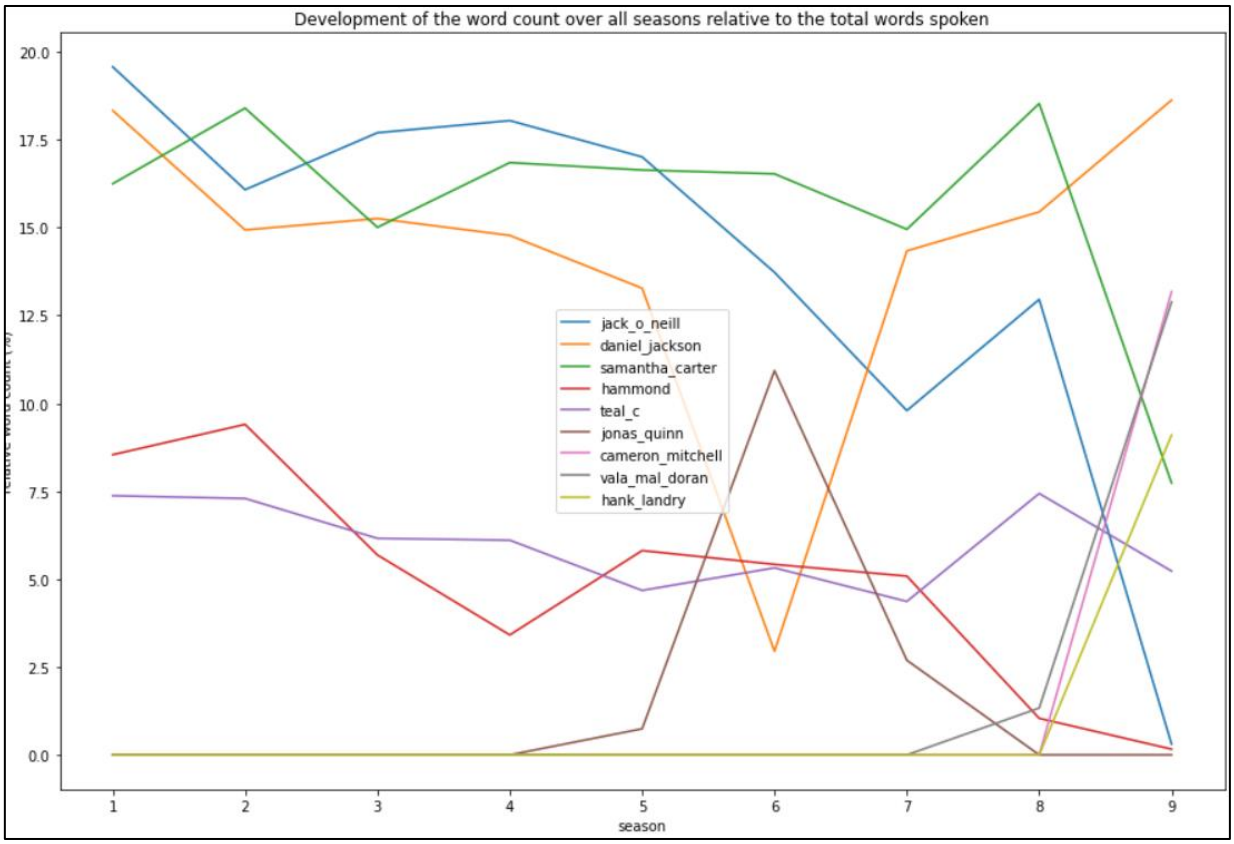


Figure 40: Development of the word count over all seasons relative to total word count

Surprisingly, the figure is nearly identical to the previous Figure 39, which shows the total number of words. That leads to the assumption that there is a direct relation between the number of episodes and the number of words spoken in total. It is interesting to see that the characters “Jack O’Neill”, “Samantha Carter”, and “Hammond” have a decrease in their word count in season 9. That suggests that these characters that are main characters in season 1 – measured by their word count in that season – may not be as important anymore in the last season, with

other characters having emerged. Those new characters are “Cameron Mitchell”, “Vala Mal Doran”, and “Hank Landry”. These characters join the cast in season eight and become main characters in season nine. They do not exist yet in seasons one to seven, as visible in the line chart, as all lines are at zero.

In summary, no matter if one regards the word count relative to the total amount of words spoken in the season or not, the charts show significant differences in the numbers. Therefore, the hypothesis that the amount of speech of a character varies significantly over the seasons can be accepted.

5.3.6 Hypothesis 5 – Sentiment of Characters

The fifth hypothesis discusses the sentiments of the characters. Since it can be assumed that there are no strong opinions in the plot as there would be, for example, in reviews, the hypothesis assumes that the general sentiment of a character tends to be neutral.

The Sentiment Analysis follows a document-based approach, where each block of text spoken by a character is considered a document. Dependent on the character’s line, this can also be only one word or sentence. For the Sentiment Analysis, we use “AFINN”, a python package that uses a wordlist-based approach to assign a sentiment score to text [50].

This score usually ranges from -5 to 5, with rare values that lie outside this interval. A negative number symbolizes a negative sentiment, while a positive number shows a positive sentiment. Neutral sentiment has a score of zero [50].

The analysis examines the five characters with the biggest word count per season. The nine characters that were examined are:

- Jack O’Neill
- Daniel Jackson
- Samantha Carter
- Teal’C

- General Hammond
- Jonas Quinn
- Cameron Mitchel
- Vala Mal Doran
- Hank Landry

For each character, the spoken text is extracted, and each line is provided with a sentiment score that is displayed in a separate column of the data frame. A function then classifies the sentiment as neutral, negative, or positive based on the score and labels it as such in a separate column. An example of the resulting tables is shown in the following Table 6. The table shows the scores and sentiments for the character “Jack O’Neill”.

	character	text	season	episode	sentiment_score	sentiment
30	jack_o_neill	retired	1	1	0.0	neutral
32	jack_o_neill	air force	1	1	0.0	neutral
34	jack_o_neill	a little piece of advice major get reassed to ...	1	1	0.0	neutral
36	jack_o_neill	never heard of him	1	1	0.0	neutral
40	jack_o_neill	ive been here before	1	1	0.0	neutral
46	jack_o_neill	retired	1	1	0.0	neutral
48	jack_o_neill	thought about it but then id have to shoot any...	1	1	1.0	positiv
50	jack_o_neill	major samuels mentioned something about the st...	1	1	0.0	neutral
55	jack_o_neill	ya think	1	1	0.0	neutral
60	jack_o_neill	weapons sir	1	1	0.0	neutral
64	jack_o_neill	yes sir there were no creatures like this on a...	1	1	2.0	positiv
66	jack_o_neill	yeah his eyes glowed that was our first clue	1	1	1.0	positiv
68	jack_o_neill	unless he can survive a tactical nuclear warhe...	1	1	2.0	positiv

Table 6: Table with sentiment of 'Jack O'Neill'

To display the sentiment of the characters, the average values of the sentiment in each episode are shown in bar charts. The x-axis of the charts is a fusion of the season and the episode number. The first digit is the season, while the second and third digits of the number are the episode. The first three characters that will be examined are Jack O’Neill, Daniel Jackson, and Samantha Carter. The corresponding bar charts are shown in Figure 42, Figure 41, and Figure 43.

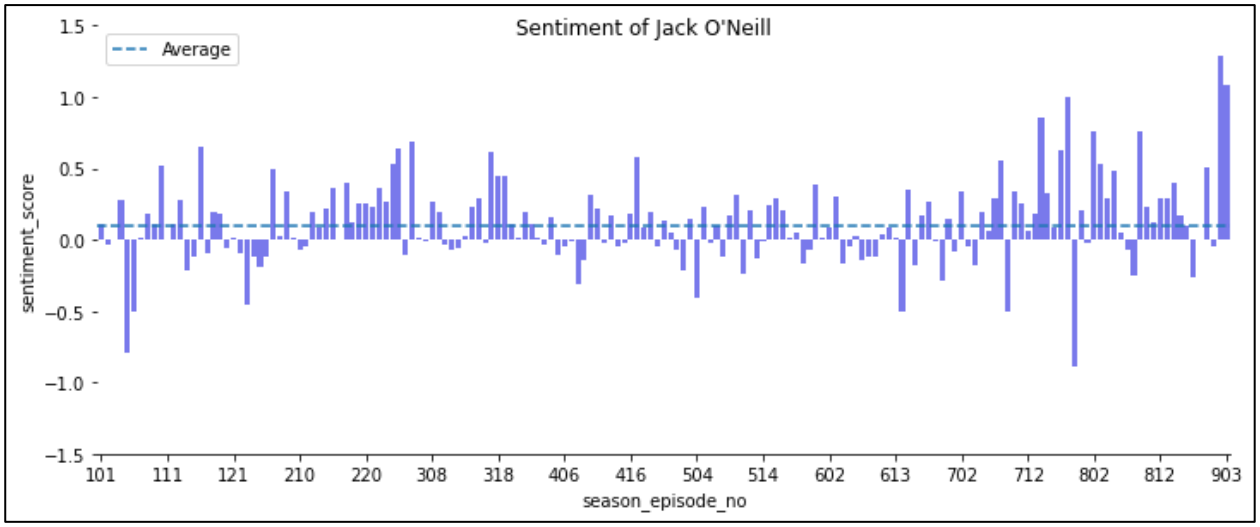


Figure 42: Sentiment over the episodes of Jack O'Neill

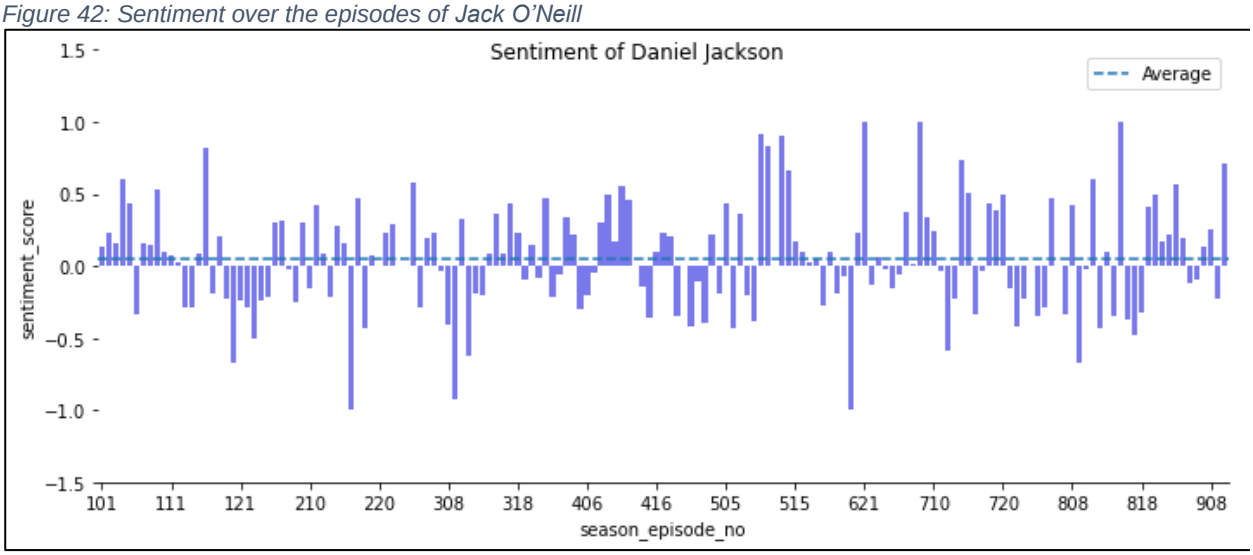


Figure 41: Sentiment over the episodes of Daniel Jackson

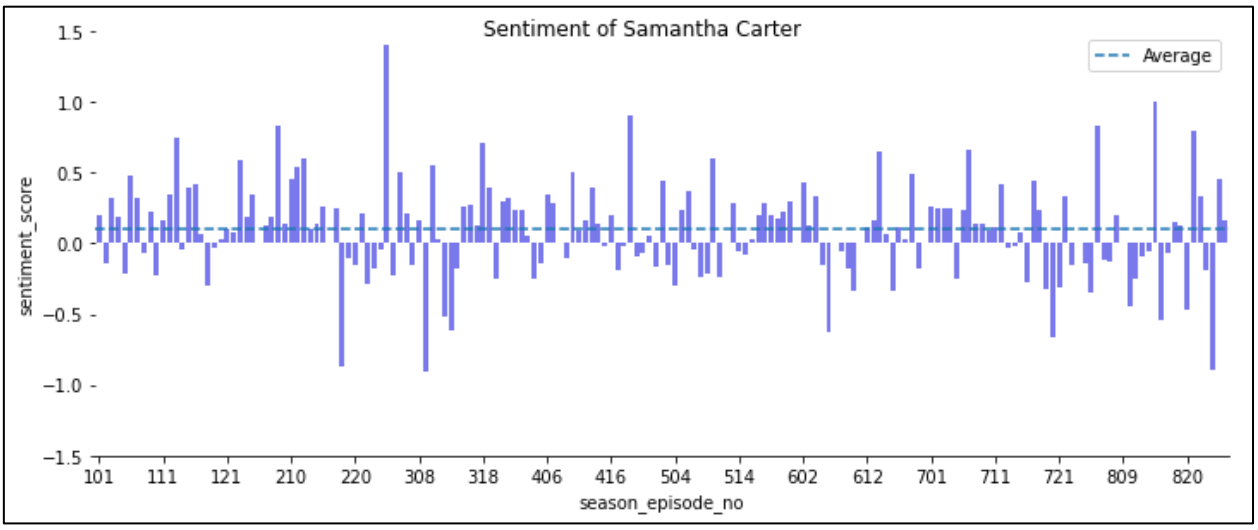


Figure 43: Sentiment over the episodes of Samantha Carter

The first look shows that the sentiments are shown in Figure 42, Figure 41 and Figure 43 vary for all three characters over the nine seasons. It seems, though, as if the sentiment is more often on the positive y-axis than the negative. Most values of episodes are still very close to zero, leading to the assumption that there are either a near equal amount of positive and negative sentiments that cancel each other out or the dialogue is mostly neutral with rare positive and negative expressions.

The characters' average sentiment scores over all seasons and episodes are close to zero but positive, as well. The following Table 7 shows the exact values for the average sentiments of the characters:

Character	Average sentiment score
Jack O'Neill	0.093
Daniel Jackson	0.052
Samantha Carter	0.097

Table 7: Average sentiment score of Jack O'Neill, Daniel Jackson & Samantha Carter

The next character is Teal'C. The bar chart that shows his sentiments over all episodes he featured in is shown in the following Figure 44.

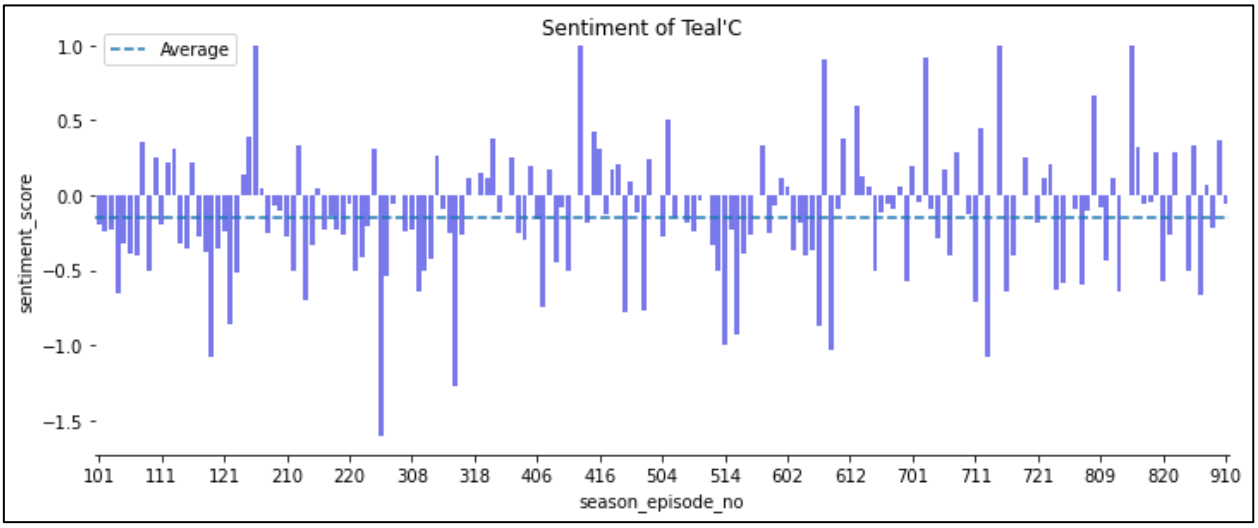


Figure 44: Sentiment over the episodes of Teal'C

There are more episodes that have a negative sentiment score. That is supported by the average score over the whole series of -0.15. The score is still very close to zero. However, Teal'C is the only inspected character with a negative average score. The tendency to a negative sentiment could come from the fact that Teal'C has rather few lines since the character mainly relies on facial expressions rather than spoken words. He does not talk much, but when he speaks, he could speak up about concerns or give warnings, thus the negative sentiment [46].

Next, we will take a look at General Hammond. Figure 45 shows his sentiment per season.

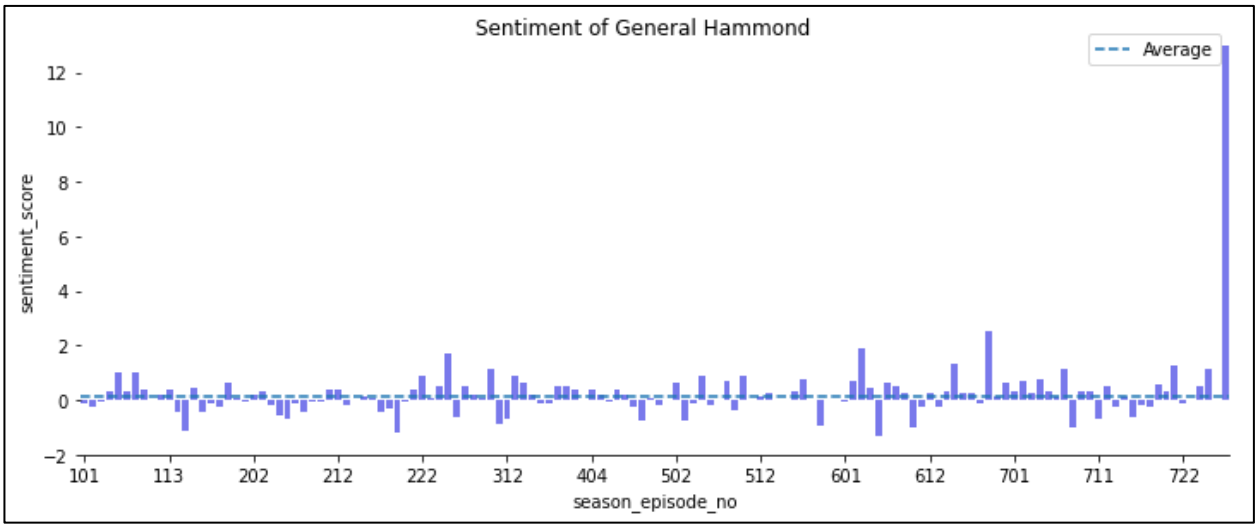


Figure 45: Sentiment over the episodes of General Hammond

The number of scores over and under the x-axis seems to be near similar, with a slight tendency to positive sentiment. General Hammond's average score over the whole series lies at 0.11. Interesting is the outlier in season 9, episode 10, where the score is 14. In that episode, Hammond speaks only once. Thus, there are no other values to relativize the score. The concrete text that he is speaking is a speech that praises the team of the main characters [51].

The next character that is analysed is Jonas Quinn. His sentiments are shown in Figure 46.

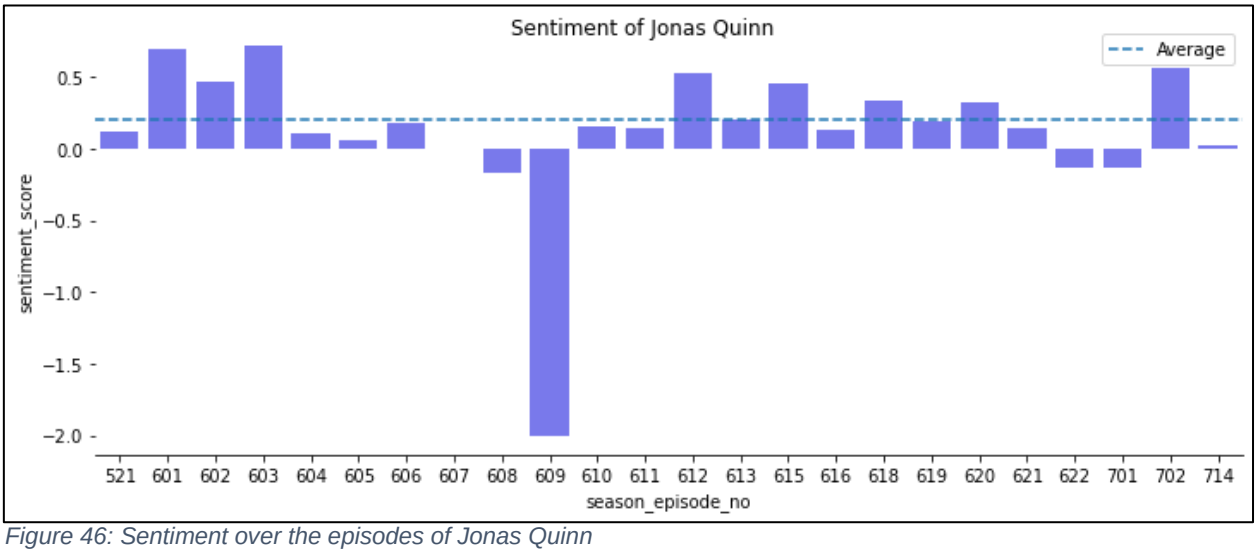


Figure 46: Sentiment over the episodes of Jonas Quinn

Jonas Quinn's sentiment is mostly on the positive y-axis. However, the values still lie between zero and one. Therefore, the positive sentiment is not very prominent. His average sentiment score is 0.21. Interesting is that he has one of the averages that is the farthest away from zero. The reason for this could be that Jonas Quinn is only part of seasons five, six and seven. Therefore, he has fewer lines than, for example, Samantha Carter, who is part of every season, so that there are fewer negative and positive sentiments that can cancel each other out. There is an outlier in season six episode nine. In this episode, Jonas Quinn speaks only one time. Thus, there are no other lines that relativize the score. In this specific part, the character is complaining, therefore being assigned a negative value [52].

The last three characters can again be grouped together, as their bar charts show similar distributions. The bar charts of Cameron Mitchell, Vala Mal Doran and Hank Landry are shown in Figure 49, Figure 48, and Figure 47.

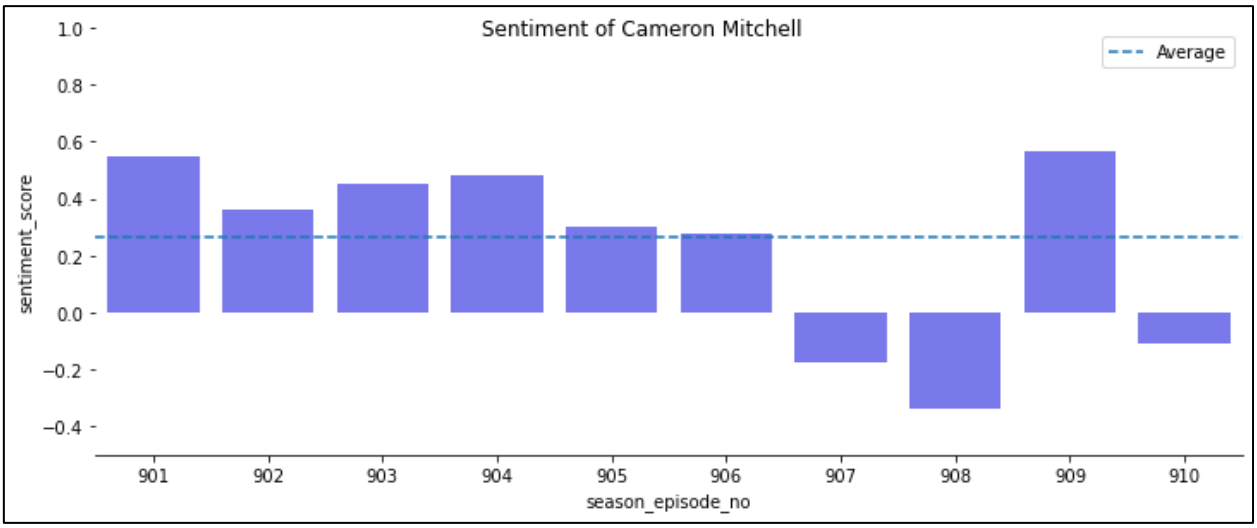


Figure 49: Sentiment over the episodes of Cameron Mitchell

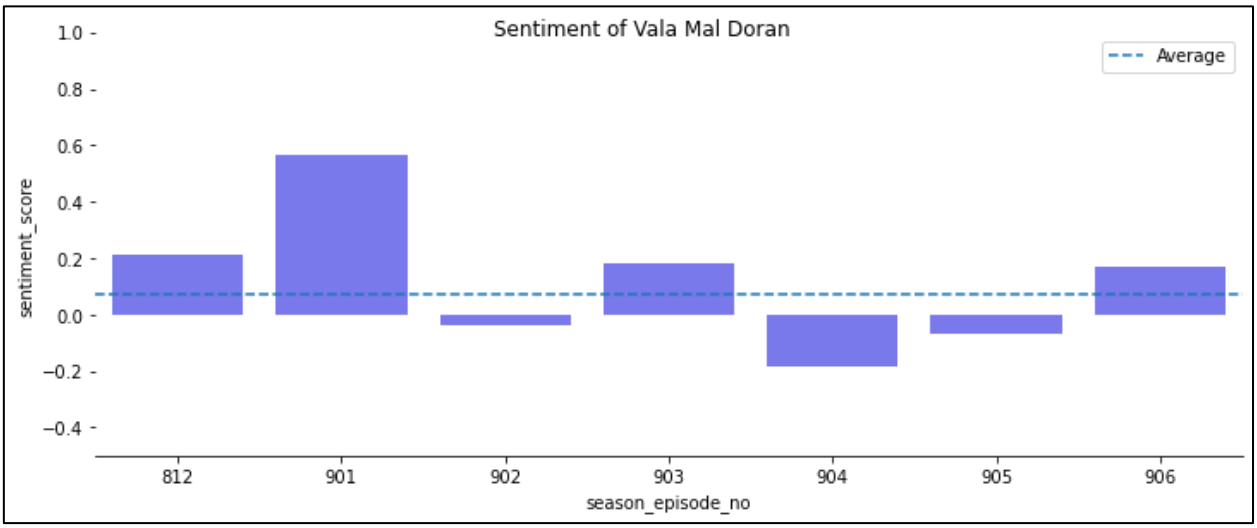


Figure 48: Sentiment over the episodes of Vala Mal Doran

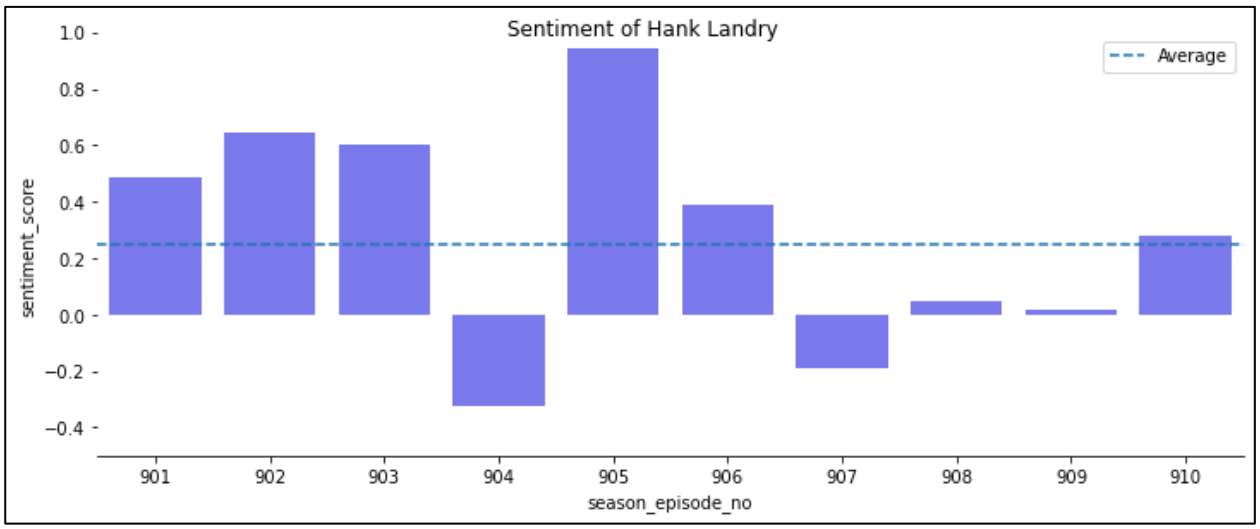


Figure 47: Sentiment over the episodes of Hank Landry

All three of these characters feature in only part of the series. The number of episodes they feature is at seven or ten. The bars in the figures are mostly on the positive scale, and they all have average sentiment scores over zero. The exact numbers are shown in the following Table 8.

Character	Average sentiment score
Cameron Mitchell	0.264
Vala Mal Doran	0.074
Hank Landry	0.246

Table 8: Average sentiment score of Cameron Mitchell, Vala Mal Doran & Hank Landry

Interesting is that both Cameron Mitchell and Hank Landry have a comparatively high average score. Both feature in relatively few episodes, so it seems to be easier to find a tendency towards a sentiment as there are fewer data points that cancel each other out. However, the character Vala Mal Doran goes against that theory. With only seven episodes she features in, she has the lowest number of episodes. However, her average score is really close to zero.

There are several possible explanations for this: First, it could just be the nature of the character to not express a sentiment with her words. Second, she does not have enough text to actually have a tendency towards negative or positive sentiment. The third and last possibility seems the most plausible one and simultaneously cancels the second possibility. Even though Vala Mal Doran features in only seven episodes, she is a main character in those episodes and therefore has a high amount of speech. Therefore, the data points relativize each other.

We now take a look at the exact number of times that a text has been classified as neutral, positive, or negative.

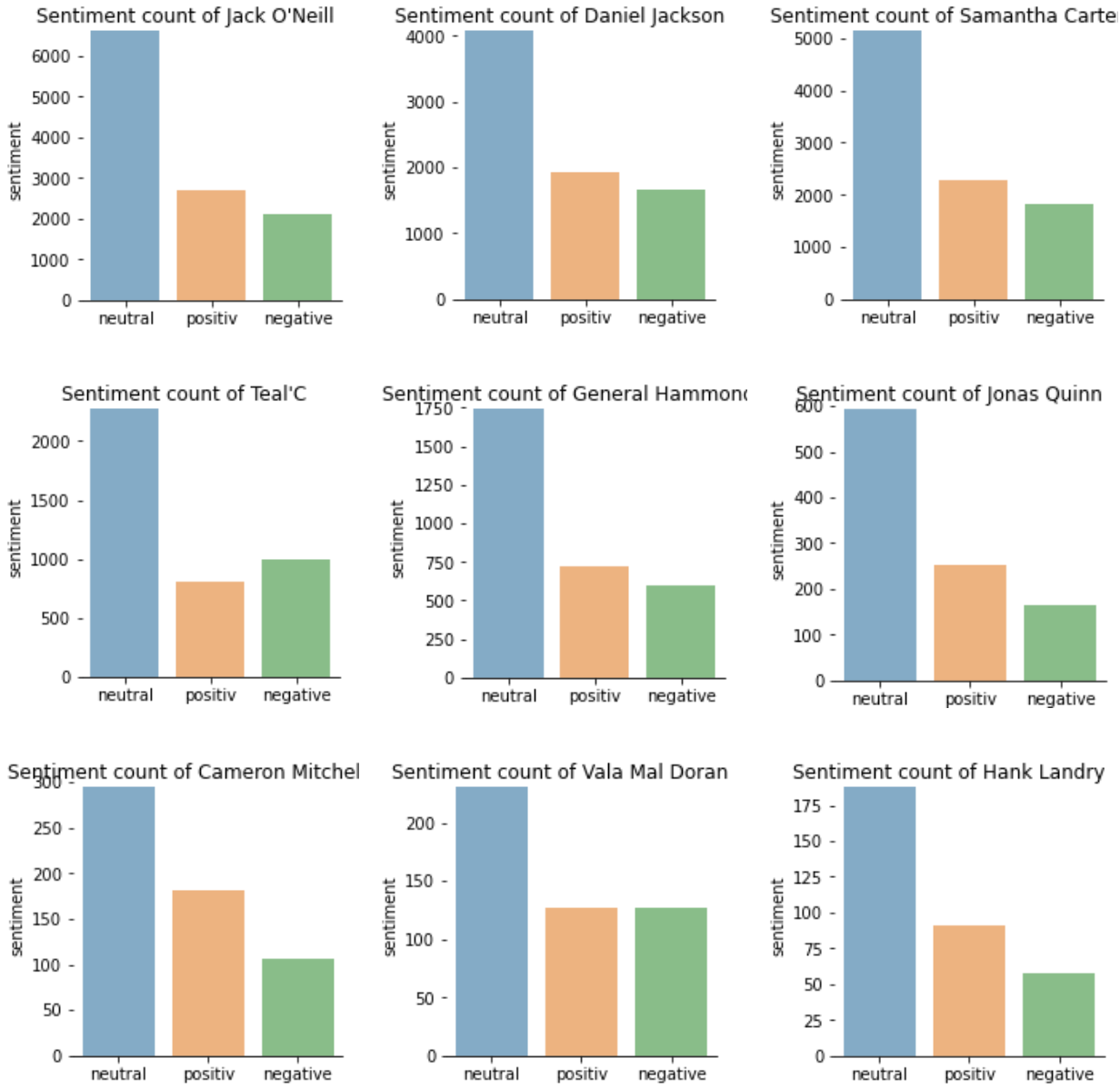


Figure 50: Number of times text was classified as neutral, positive, or negative

Figure 50 shows how often each character's sentiments have been classified as neutral, positive, or negative. The neutral sentiments far surpass the amount of positive or negative sentiment for each character. It varies if the character has more negative or more positive text, but the neutral sentiment doubles the counts of the other sentiments for all but two characters. Even for those characters, the count of neutral sentiments is still much higher.

Before we draw a conclusion about the hypothesis, it is important to note that the sentiment classifier has certain boundaries. The classifier is based purely on the written words, without

context or knowledge about the tone of the word is said within the series. Linguistic means such as sarcasm or irony can thus not be detected. Expressions that are objectively neutral but still carry a sentiment are also very difficult to detect [34].

An example where the classifier failed to detect the sentiment is a piece of a dialogue sequence in the series in episode one of the first season.

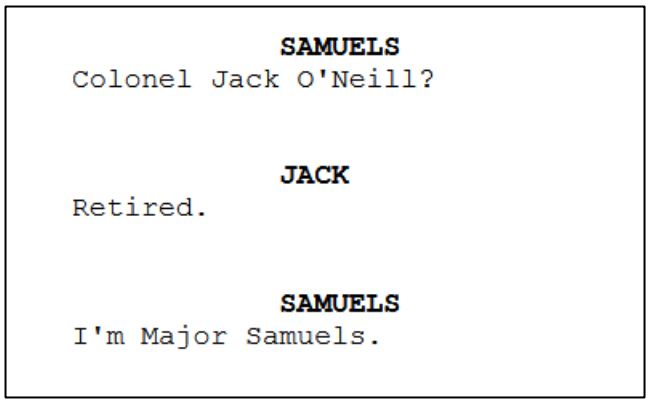


Figure 51: Extract of a dialogue from season 1 episode 1 [53]

Figure 51 shows an excerpt of this dialogue. The important part here is the answer "Retired" that Jack O'Neill – in Figure 51 dubbed as simply "Jack" – returns to the dialogue partner. The classifier classifies this with a score of zero, therefore labelling it as neutral. According to the expert's knowledge, the viewer of this scene clearly notices that this response is said in a rather unfriendly tone, therefore considering it negative [46].

In general, especially the character Jack O'Neill often uses sarcasm [46]. Thus, the classification of his text might be inaccurate in certain parts.

In total, though, regarding the bar charts, the average scores and the sentiment counts of the nine characters that have been analysed, there is no reason found that would cause the hypothesis to be rejected. The hypothesis that the general sentiment of a character tends to be neutral can therefore be accepted.

5.3.7 Overview Generation

Based on the previous hypotheses, the goal is to create an overview of a season or episode that displays – after specifying the season and, if desired, the episode – the number of characters in the season or episode and a list of them, the three characters with the highest word count (in relation to Hypothesis 1 – Determination of , these characters can be assumed to be main characters), the total amount of words spoken, a graph that shows the sentiment in each spoken line and the distribution of the topics in the episode or series.

The overview over the season additionally shows a line chart with the average word count of the three characters with the most words spoken to examine any developments.

The episode and season are specified by using the number as input into the function generate_overview_episode(season_no, episode_no)(Attachment C). The first input field is the season. The second is the episode.

The following Figure 52 shows an example overview over season one, episode one.

Figure 52: Generated Overview of S1E1

The overview for a season is generated similarly by calling the function generate_overview_season(season_no)(Attachment B). The input parameter specifies the season. The output is the same as for the episode overview. However, the overview has an additional graph showing the average amount of speech over the episodes of the season of the three characters with the biggest amount of speech.

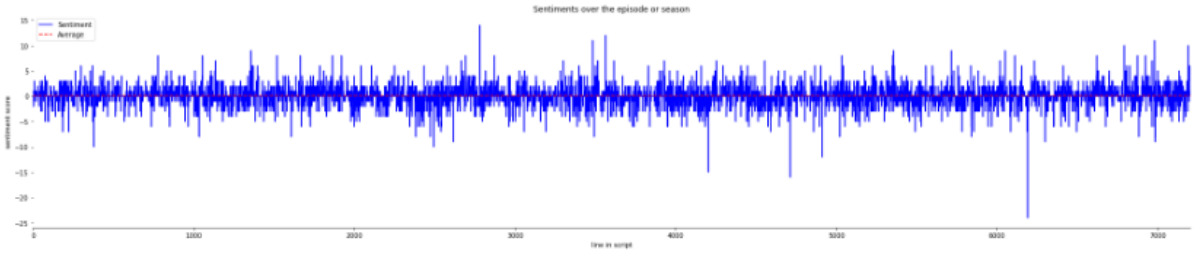
An example overview of season 1 is shown in the following Figure 53.

Shown is an overview over season 1

There are 106 characters in the season.
These characters are airman, woman, officer, apophis, soldier, samantha_carter, jack_o_neill, hammond, warner, kawalsky, harriman, ferretti, daniel_jackson, skaara, sha're, teal_c, bo'la, boy, tech, medic, warren, native, goa'uld, casey, dr langford, scientist, catherine, martha, earnest, dr_janet, priest, bra'tac, o`neill, drey'auc, rya'c, nem, mackensie, kleinhouse, cole, hathor, solder, cassie, nurse, dr warner, davis, hanno, man, young hanno, villager, shak'l, narim, omoc, tuplo, maybourne, lya, siler, harlan, tv reporter, soldier 2, solder 3, airmen, jaffa, "auto destruct in, walter, doctor, kennedy, kinsey, entity, voice, ml, klorel, abu, ., mughal, turghan, nya, guard, makepeace, marine, leedora, johnson, connor, franks, baker, hanson, jamala, sara, dad, reporter, sara's dad, crystal, answering machine, police officer, charlie, secretary, nefrayu, oper, anteaus, alekos, thetyes, kynthia, argosian, gairwyn, thor, kendra, and unas.

The 3 with the highest word counts are jack_o_neill, daniel_jackson, and samantha_carter.

In total, there are 76413 words spoken.
The sentiment score over the episode/season has an average of 0.0303.
The graph below shows the sentiment for each line in chronological order.



The amount of speech each of the top 3 characters has per episode, is as follows:

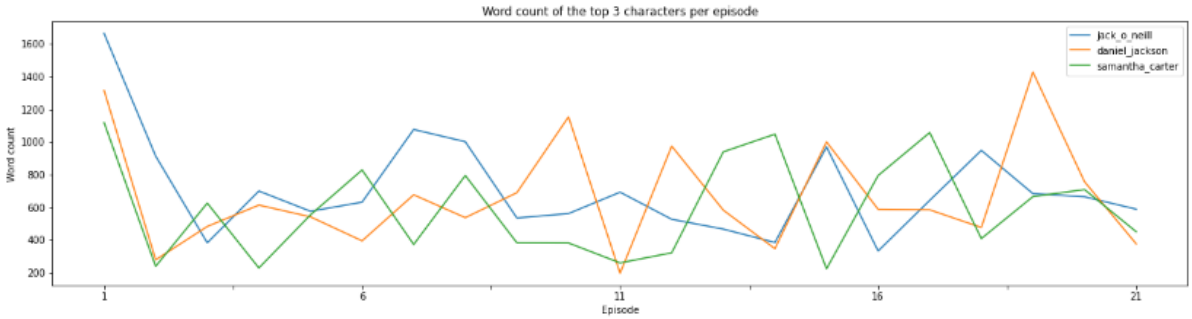


Figure 53: Generated Overview of S1

In case of the specified episode or season not existing, a corresponding notice is displayed.

6. Conclusion

This chapter gives concluding thoughts concerning the whole project. The following sections contain a Summary, a Critical Reflection and an Outlook.

6.1 Summary

One can find numerous reviews and summaries of movies and series with different opinions on the internet. However, their content differs from author to author, as the texts are subjective and change depending on the author's opinion. This causes these reviews or summaries to differ from the actual content. In order to make an informed decision about a movie or series, there needs to be an objective instance that can evaluate the movie or series.

This paper's goal is it to use Natural Language Processing to process movie or series scripts and analyse the plot content, characters, and their evolution.

After a short Introduction in chapter 1, the problem statement formulates five hypotheses and requirements for a computer-generated overview of a user-specified episode or season in chapter 2 "Problem Statement".

Then, to create a basic understanding of the discussed topics, the next chapter 3 "Theoretical Fundamentals", explains the theory. The topics discussed are Natural Language Processing (chapter 3.1), Text Pre-Processing Steps in Natural Language Processing (chapter 3.2), Topic Modelling (chapter 3.3), Sentiment Analysis (chapter 3.4) and lastly the State of the Art (chapter 3.5).

Following the Theoretical Fundamentals, is a walk-through of the paper's Methodology in chapter 4. As the paper uses the CRISP-DM, its processes and aspects are explained in this chapter.

The first step here is Data Acquisition, discussed in chapter 5.1. This chapter explains the structure of the data and covers how to attain the script links and parse the script code. Chapter 5.2 "Data Preparation", gives an Exploratory Data Analysis and discusses data cleaning. Lastly,

chapter 5.3 “Modelling”, examines the five formulated hypotheses and contains the overview generation.

Hypothesis one assumes that the main characters can be detected through the amount of speech and must be rejected since a main character has to speak to be detected as such. Several approaches to differentiate main characters from side characters were examined. However, each of them fails to recognize main characters that communicate mostly through facial expressions and body language.

The second hypothesis assumes that the topics detected in the script are science-fiction topics, as the examined series “Stargate” is considered a science-fiction series. This hypothesis has to be rejected as well, since the detected topics were labelled as *“Negotiation”, “War”, “Faith”, “Sci-fi and injury” and “Fighting”*.

Hypothesis three assumes that men speak more than women and that these two genders discuss different topics. The first part can be accepted. The examinations showed that men speak about twice as much as females, with only a few outliers that can be attributed to the plot of specific episodes. However, the second part must be rejected since the distribution over the four detected topics is approximately the same for men and women.

The fourth hypothesis assumes that the amount of speech a character has varies significantly over the seasons and is accepted after exemplarily observing the most speaking characters in the seasons.

Lastly, the fifth hypothesis claims that the general sentiment of a character tends to be neutral and can be accepted after examining nine main characters as examples. The average sentiment of these nine characters is close to zero without exceptions, and the count of how many lines were classified as neutral exceeds that of both positive and negative sentiments.

Finally, the paper presents a method to generate an overview of an episode or season that contains the aforementioned areas of interest. Therefore, the overview includes the number of characters in the season or episode and a list of them, the three characters with the highest word count, the total amount of words spoken and a graph that shows the sentiment in each spoken line, as well as the distribution over the five identified topics of the season or episode.

The overview over the season shows an additional line chart with the average word count per episode of the season of the three characters with the highest word count to examine any developments.

6.2 Critical Reflection

Although we were able to successfully examine all hypotheses, some critical aspects have to be noted.

Due to the open-source nature of the IMSDb website that was used to obtain the scripts for this project, the data itself has some deficits. Firstly, as the scripts were transcribed by different persons on a non-profit, voluntary basis, there are differences in how certain series-specific words or names are written. For example, one transcriber uses abbreviations to refer to characters, while most other transcribers write the whole name. A concrete example here would be the character “General Hammond”, who throughout the scripts is referred to as “General Hammond”, “Hammond”, or “GH”. Additionally, spelling mistakes like “Hamond” or “Hamnond” occur.

In general, the website does no additional checks on the uploaded scripts. Hence, spelling errors or grammar mistakes occur. For the same reason as transcribing on a voluntary basis, the last season of the series “Stargate SG-1” has no scripts accessible via IMSDb. Statements regarding the whole series thus have to be taken with a grain of salt.

Considering the pre-processing of the data, not all versions of writing a character’s name might have been considered and spelling mistakes could have caused a word to be falsely normalized. Due to the manual configuration of the normalizer, some mistakes or aspects may not have been taken into account. Series-specific words are also difficult to process by the normalizer. The term “Ja’ffa” that is used in the series, for example, was fragmented into “Ja” and “ffa”. Additionally, the data cleaning only used unigrams, so terms that consist of two words like “black hole” are separated. Thus bi-grams or tri-grams should have been included in the data cleaning process.

Next to the general critics of the data in itself, certain aspects during the examinations of the hypotheses must be considered.

Firstly, during the examination of the first hypothesis concerning the detection of main characters, the thresholds of the approaches that were proposed to detect main characters were determined through testing. They depend on the specific data of this series. More research would be needed to see if there is an objective way to determine those values.

The same is true for the hyperparameters used for the models of the second hypothesis. This hypothesis used models to determine the topics within the series. Due to limited computing power, there was only a limited range of values tested to determine the best hyperparameters. Further, the metrics used to evaluate the topic models are not ideal. However, the lack of cheap and easily usable metrics is a general problem of topic modelling. Additionally, during the examination, Latent Dirichlet Allocation (LDA) was used to model the topics. LDA performs well for long texts. However, the method of organizing the data for the modelling process was changed from examining episodes to examining scenes. The scenes are significantly shorter. Thus, the approach of Non-negative Matrix Factorizing (NMF) should have been considered, as it performs better on shorter texts.

Furthermore, to reach a better differentiation when modelling the topics, common words with low significance were removed. This makes it harder to find a label for a topic. Therefore, there is a trade-off between finding topics that are easily interpretable and distinct from each other.

The third hypothesis was separated into two parts. The first one examines the differences in the amount of speech between male and female characters. The second examines if men and women talk about different topics. Both parts share the problem that some names of characters could not be assigned a gender. Some names were unique due to the alien nature of the character or otherwise not identifiable because of the use of abbreviations or merely using a job title like “Doctor”. Further, some characters with different genders shared the same name.

As the second part includes topic modelling, the same critics made for hypothesis 2 apply here.

The fourth hypothesis examines the word count of characters. Due to limitations of time and computing power, only the main characters of each season were examined. Therefore, one can not conclude that the hypothesis also holds true for side characters. Further examinations are needed here.

The fifth hypothesis inspects the sentiment of the characters. To perform the Sentiment Analysis, the classifier “AFINN” was used. This classifier is free of charge and easily usable. However, it does not normalize the values. Thus, strong outliers are possible. Other existing classifiers were not tested, and further investigation is necessary to determine if a different classifier would be better suited. Additionally, due to limitations of time, again, only the main characters were examined.

The Sentiment Analysis itself has certain drawbacks. Firstly, as of now, there is no way to detect sarcasm and irony in written text. In general, Sentiment Analysis has no regard for visual or aural criteria that would normally be used by a human to determine sentiment.

Lastly, the overview is based on all hypotheses and the therefore developed algorithms. Thus, the same problems and critics that apply to the hypotheses also apply to the generation of the overview.

6.3 Outlook

Considering the Summary and Critical Reflection, we want to give an outlook on the future of this project. As stated in the Introduction to this paper, the goal was to find a way to give an objective insight into a series or movie. With this work, this is now possible.

In the previous chapter, “Critical Reflection”, we already discussed some points that can be improved. Further works regarding this project can touch on those deficits. For example, a different method of topic modelling can be tested, as well as increasing the number of characters that were regarded. Further, to increase the usability of this work, the display of the overview can be adapted, for example, by creating a dashboard.

To add more aspects and data to the investigation of the series “Stargate SG-1”, the movies preceding and following the plot of the series can be included in the research. Furthermore, a comparison can be added by examining series whose plots happen in the same fictional universe as “Stargate SG-1”.

In general, Natural Language Processing is a field of study that is still constantly evolving. Further developments, for example, in the area of detecting sarcasm or irony and developing better topic model metrics, can be expected. Those new insights can then be integrated into this work.

7. References

References

[1] IMDb, *Press Room - IMDb.* [Online]. Available: https://www.imdb.com/pressroom/stats/ (accessed: Mar. 15 2022).

[2] IMDb, *The Shawshank Redemption (1994) - IMDb.* [Online]. Available: https://www.imdb.com/title/tt0111161/reviews?ref_=tt_urv (accessed: Mar. 15 2022).

[3] IMDb, *Stargate SG-1 (TV Series 1997–2007) - IMDb.* [Online]. Available: https://www.imdb.com/title/tt0118480/reviews?ref_=tt_urv (accessed: Mar. 15 2022).

[4] IMDb, *User-submitted review of "Stargate SG-1&quot.* [Online]. Available: https://www.imdb.com/review/rw1543961/?ref_=tt_urv (accessed: Mar. 15 2022).

[5] IMDb, *User-submitted review of "Stargate SG-1&quot.* [Online]. Available: https://www.imdb.com/review/rw1887419/?ref_=tt_urv (accessed: Mar. 15 2022).

[6] S. Ghosh and D. Gunning, *Natural language processing fundamentals: Build intelligent applications that can interpret the human language to deliver impactful results / Sohom Ghosh and Dwight Gunning*. Birmingham: Packt Publishing, 2019.

[7] D. Khurana, A. Koli, K. Khatter, and S. Singh, “Natural Language Processing: State of The Art, Current Trends and Challenges,” Aug. 2017. [Online]. Available: https://arxiv.org/pdf/1708.05148

[8] A. Kulkarni and A. Shivananda, *Natural language processing recipes: Unlocking text data with machine learning and deep learning using Python*. New York: Apress, 2021.

[9] P. Koehn, *Statistical Machine Translation*: Cambridge University Press, 2009.

[10] F. Sebastiani, “Machine learning in automated text categorization,” *ACM Comput. Surv.*, vol. 34, no. 1, pp. 1–47, 2002, doi: 10.1145/505282.505283.

[11] Y. Yang and X. Liu, “A re-examination of text categorization methods,” in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '99*, New York, New York, USA, 1999.

[12] I. Mani, *Automatic Summarization*. Philadelphia: John Benjamins Publishing Company, 2001.

[13] S. Arora, K. Batra, and S. Singh, *Dialogue System: A Brief Review*: arXiv, 2013.

[14] N. Indurkhya, *Handbook of Natural Language Processing, Second Edition,* 2nd ed. Hoboken: Taylor and Francis, 2012. [Online]. Available: http://gbv.eblib.com/patron/FullRecord.aspx?p=565922

[15] R. Mitkov, *The Oxford handbook of computational linguistics*. Oxford: Oxford Univ. Press, 2009.

[16] E. Mays, F. J. Damerau, and R. L. Mercer, “Context based spelling correction,” *Information Processing & Management*, vol. 27, no. 5, pp. 517–522, 1991, doi: 10.1016/0306-4573(91)90066-U.

[17] A. G. Jivani, *A comparative study of stemming algorithms*, 2011. [Online]. Available: https://kenbenoit.net/assets/courses/tcd2014qta/readings/jivani_ijcta2011020632.pdf

[18] B. Mohit, “Named Entity Recognition,” in *Theory and Applications of Natural Language Processing, Natural language processing of semitic languages*, I. Zitouni, Ed., Berlin, Heidelberg: Springer, 2014, pp. 221–245.

[19] A. Mikheev, M. Moens, and C. Grover, “Named Entity recognition without gazetteers,” in *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics* -, Morristown, NJ, USA, 1999.

[20] T. Hofmann, “Probabilistic Latent Semantic Analysis,” Jan. 2013. Accessed: May 30 2022. [Online]. Available: http://arxiv.org/pdf/1301.6705v1

[21] Nano Net Technologies Inc., *Topic Modeling with LSA, PLSA, LDA & lda2Vec.* [Online]. Available: https://nanonets.com/blog/topic-modeling-with-lsa-plsa-lda-lda2vec/ (accessed: May 30 2022).

[22] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet Allocation,” *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003. [Online]. Available: https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

[23] Cambridge University Press, *latent.* [Online]. Available: https://dictionary.cambridge.org/de/worterbuch/englisch/latent (accessed: Mar. 15 2022).

[24] T. Shi, K. Kang, J. Choo, and C. K. Reddy, “Short-Text Topic Modeling via Non-negative Matrix Factorization Enriched with Local Word-Context Correlations,” in *Proceedings of the 2018 World Wide Web Conference*, Lyon, France, 2018, pp. 1105–1114. Accessed: May 23 2022.

[25] S. George and S. Vasudevan, “Comparison of LDA and NMF Topic Modeling Techniques for Restaurant Reviews,” *03*, vol. 10, no. 62, pp. 28210–28216, 2021. [Online]. Available: https://www.researchgate.net/publication/350236296_Comparison_of_LDA_and_NMF_Topic_Modeling_Techniques_for_Restaurant_Reviews

[26] H. M. Wallach, I. Murray, R. Salakhutdinov, and D. Mimno, “Evaluation methods for topic models,” in *Proceedings, twenty-sixth international conference on machine learning*, Montreal, Quebec, Canada, 2009, pp. 1–8.

[27] M. Röder, A. Both, and A. Hinneburg, “Exploring the Space of Topic Coherence Measures,” in *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, Shanghai China, 02022015, pp. 399–408.

[28] C. Sievert and K. E. Shirley, “LDAvis: A method for visualizing and interpreting topics,” in *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, Baltimore, Maryland, USA, pp. 63–70. Accessed: Mar. 26 2022. [Online]. Available: https://nlp.stanford.edu/events/illvi2014/papers/sievert-illvi2014.pdf

[29] P. Smyth, M. Welling, and A. Asuncion, “Asynchronous Distributed Learning of Topic Models,” in *Advances in Neural Information Processing Systems*, 2008. Accessed: May 30 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2008/file/06997f04a7db92466a2baa6ebc8b872d-Paper.pdf

[30] Jonathan Chang and Jordan Boyd-Graber and Chong Wang and Sean Gerrish and David M. Blei, “Reading Tea Leaves: How Humans Interpret Topic Models,” in *Advances in neural information processing systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009 ; December 7 - 10, 2009, Vancouver, B.C., Canada ; [proceedings*, Red Hook, NY: Curran, 2010, pp. 288–296. Accessed: Mar. 26 2022. [Online]. Available: http://umiacs.umd.edu/~jbg//docs/nips2009-rtl.pdf

[31] GitHub, *Not being able to replicate coherence scores from paper · Issue #13 · dice-group/Palmetto.* [Online]. Available: https://github.com/dice-group/Palmetto/issues/13#issuecomment-371553052 (accessed: May 24 2022).

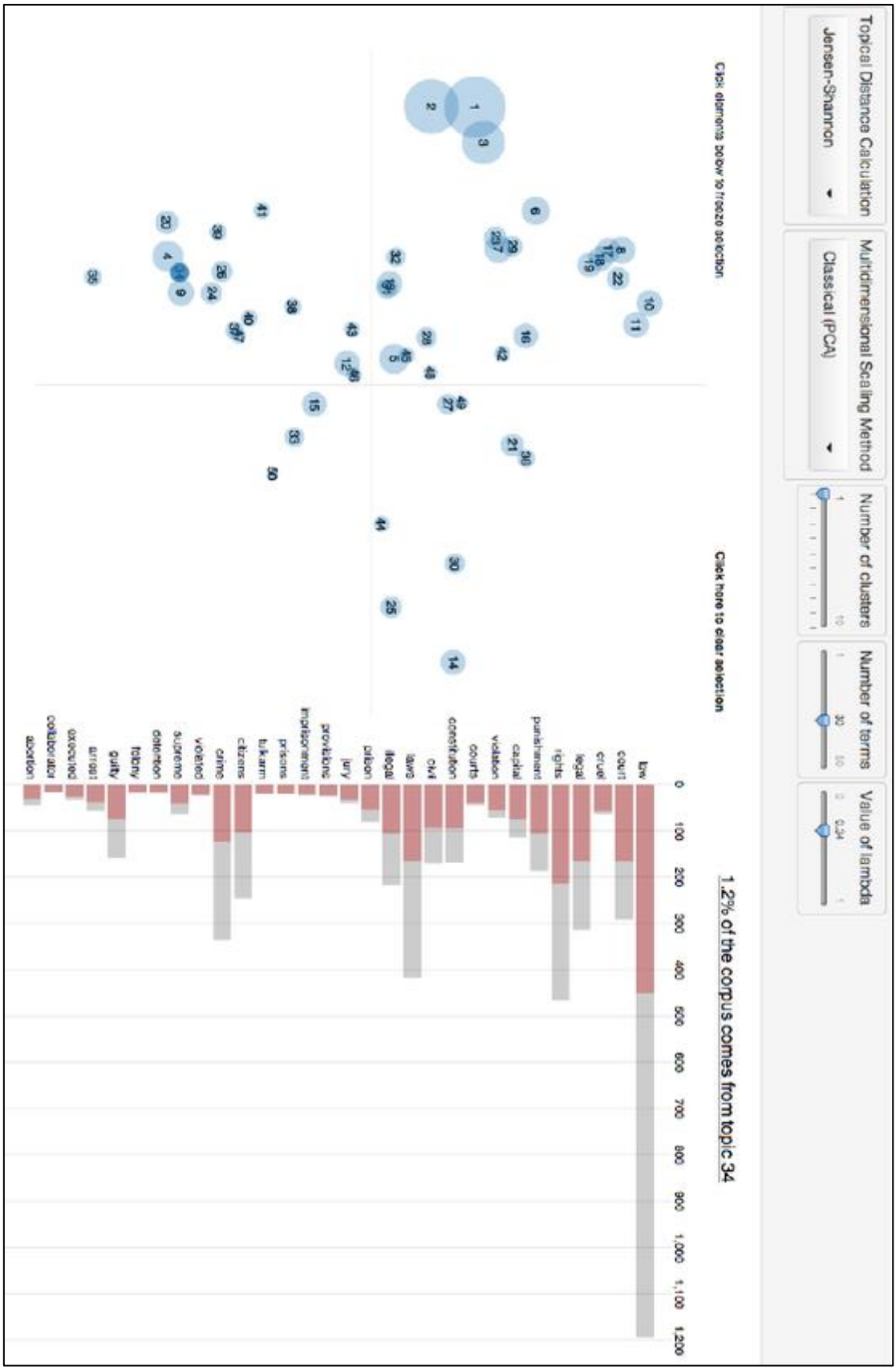
[32] A. Hoyle, P. Goel, D. Peskov, A. Hian-Cheong, J. Boyd-Graber, and P. Resnik, “Is Automated Topic Model Evaluation Broken?: The Incoherence of Coherence,” Jul. 2021. [Online]. Available: https://arxiv.org/pdf/2107.02173

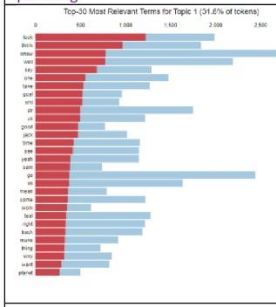
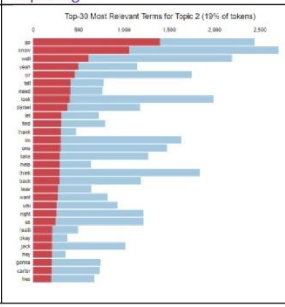
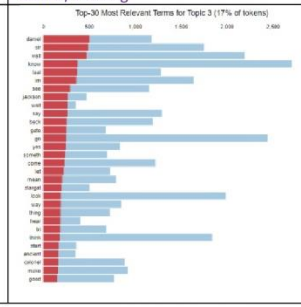
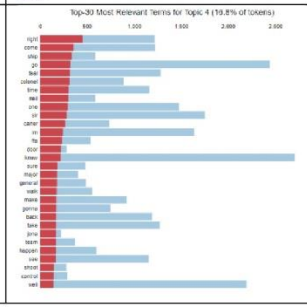
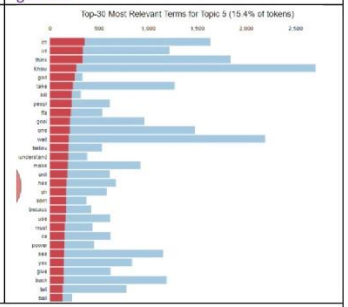
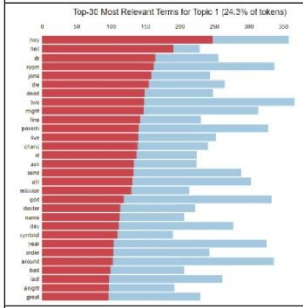
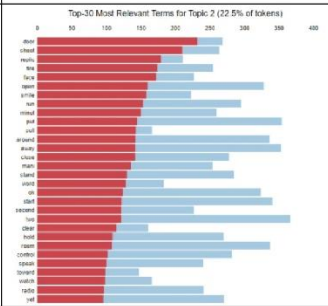
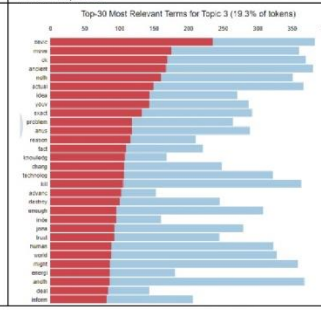
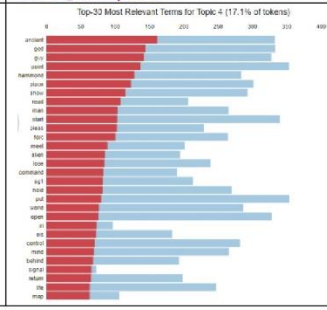
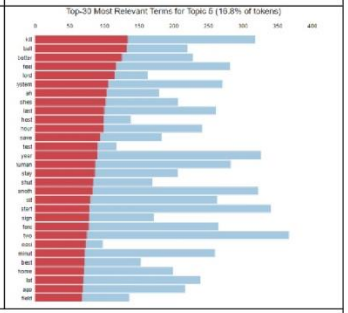
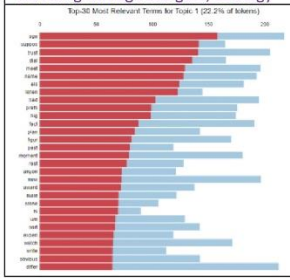
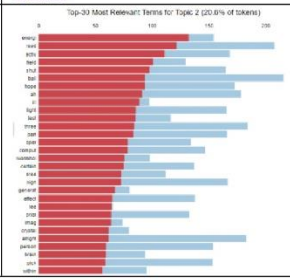
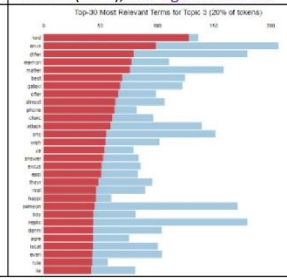
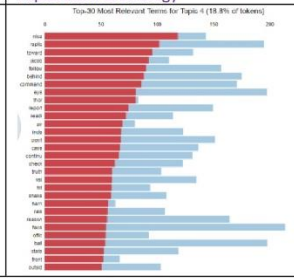
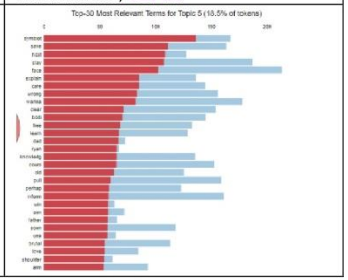
[33] A. Müller and S. Guido, *Introduction to Machine Learning with Python: Guide for Data Scientists,* 1st ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472., United States of America.: O'Reilly Media, Inc., 2016.
[34] R. Feldman, "Techniques and applications for sentiment analysis," *Commun. ACM*, vol. 56, no. 4, pp. 82–89, 2013, doi: 10.1145/2436256.2436274.
[35] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014, doi: 10.1016/j.asej.2014.04.011.
[36] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, p. 1095, 2014, doi: 10.1016/j.asej.2014.04.011.
[37] K. S. Jones, "Natural Language Processing: A Historical Review," in *Current Issues in Computational Linguistics: In Honour of Don Walker*: Springer, Dordrecht, 1994, pp. 3–16. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-585-35958-8_1
[38] Sethunya R Joseph, Kutlwano Sedimo, Freeson Kaniwa, Hlomani Hloman, and Keletso Letsholo, *Natural Language Processing: A Review*, 2016. [Online]. Available: https://www.researchgate.net/profile/sethunya-joseph/publication/309210149_natural_language_processing_a_review
[39] A. Orr, "Capital One Launched A Natural Language Chatbot Named Eno," *The Mac Observer*, 10 Mar., 2017. https://www.macobserver.com/analysis/capital-one-natural-language-chatbot-eno/ (accessed: Apr. 1 2022).
[40] Capital One, *Eno, your Capital One assistant.* [Online]. Available: https://www.capitalone.com/digital/eno/ (accessed: Apr. 1 2022).
[41] T. A. Runkler, *Data Mining: Modelle und Algorithmen intelligenter Datenanalyse,* 2nd ed. Wiesbaden: Springer Fachmedien Wiesbaden, 2015.
[42] C. Shearer, "The CRISP-DM Model: The New Blueprint for Data Mining," *Journal of Data Warehousing*, vol. 05, no. 04, pp. 13–22, 2000.
[43] IMSDb, *The Internet Movie Script Database (IMSDb).* [Online]. Available: https://imsdb.com/ (accessed: Nov. 25 2021).
[44] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," *Int. J. Mach. Learn. & Cyber.*, vol. 1, 1-4, pp. 43–52, 2010, doi: 10.1007/s13042-010-0001-0.
[45] Raindance, *The 10 Key Rules of Writing for TV.* [Online]. Available: https://raindance.org/the-10-key-rules-of-writing-for-tv/ (accessed: Jan. 12 2022).
[46] "Expert's Knowledge of Lea Vergari: Obtained through watching the episodes and reading the scripts,"
[47] Sky.de, *Neu auf Sky: Stargate: SG1 - alle Staffeln auf Abruf.* [Online]. Available: https://www.sky.de/programm-entdecken/mediathek/videos-neuaufsky/6252704228001 (accessed: May 17 2022).
[48] PyPI, *gender-guesser.* [Online]. Available: https://pypi.org/project/gender-guesser/ (accessed: Feb. 23 2022).
[49] René Raule, *Stargate Wiki – Das deutschsprachige Stargate-Lexikon.* [Online]. Available: http://stargate-wiki.de/wiki/Hauptseite (accessed: Feb. 23 2022).
[50] F. Årup Nielsen, "A new ANEW: evaluation of a word list for sentiment analysis in microblogs," *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages*, CEUR Workshop Proceedings, Volume 718, pp. 93–98, May. 2011.
[51] IMSDb, *The Fourth Horseman Transcript at IMSDb.* [Online]. Available: https://imsdb.com/transcripts/Stargate-SG1-The-Fourth-Horseman.html (accessed: Mar. 4 2022).
[52] IMSDb, *Allegiance Transcript at IMSDb.* [Online]. Available: https://imsdb.com/transcripts/Stargate-SG1-Allegiance.html (accessed: Mar. 4 2022).
[53] IMSDb, *Children Of The Gods Transcript at IMSDb.* [Online]. Available: https://imsdb.com/transcripts/Stargate-SG1-Children-Of-The-Gods.html (accessed: Feb. 25 2022).

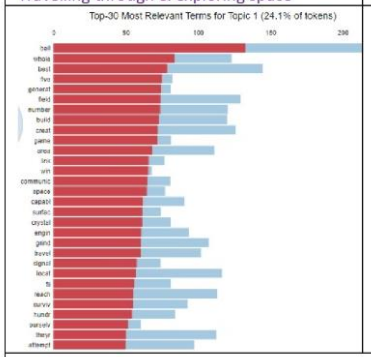
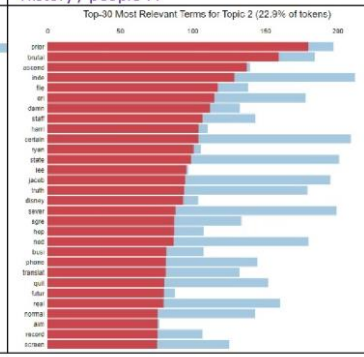
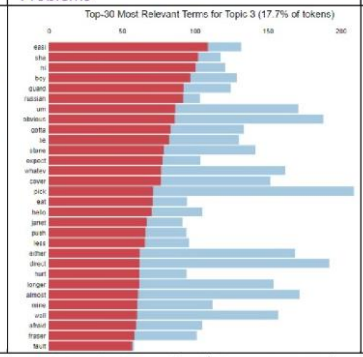
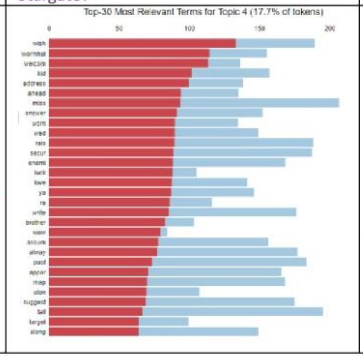
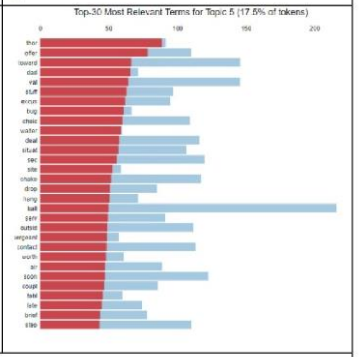
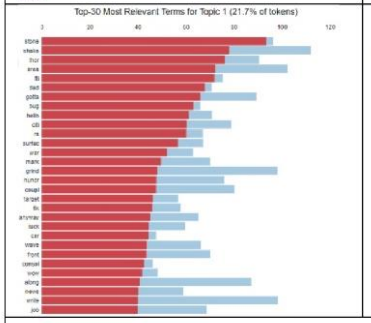
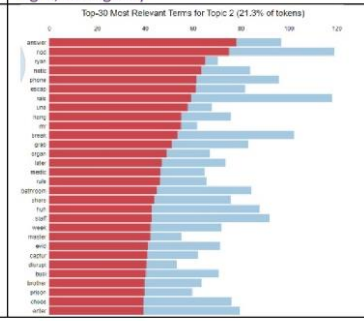
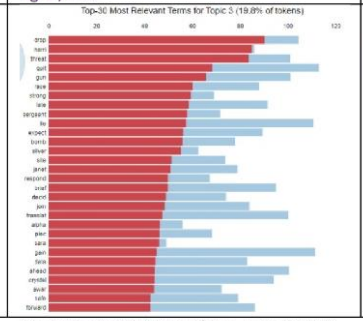
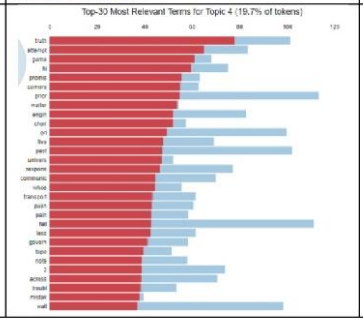
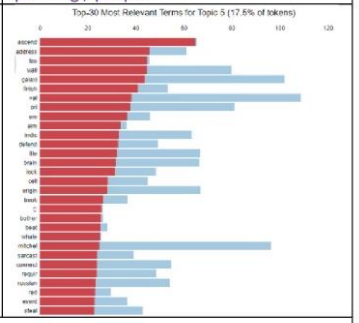
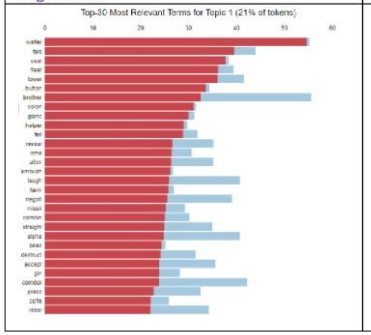
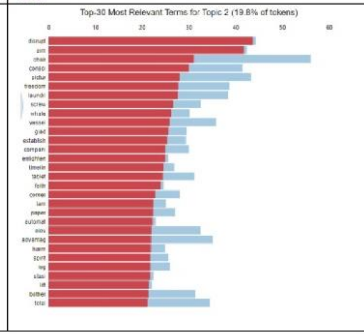
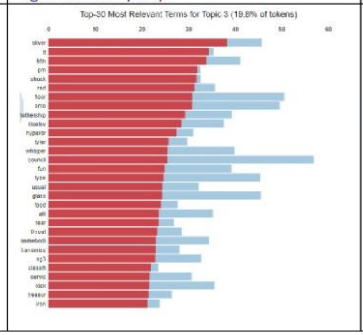
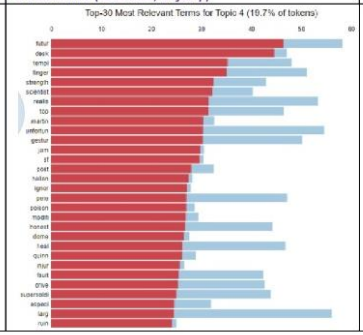
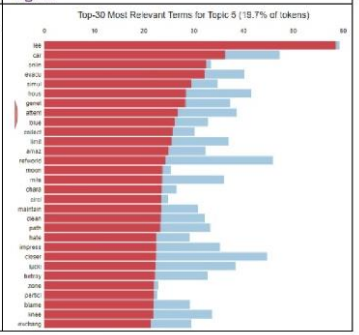
8. Attachments

- A. LDAvis Example
- B. LDA Topic Model Labelling and Comparison
- C. Overview Generation of an Episode
- D. Overview Generation of a Series

A. LDAvis Example



Model 0: tfidf 0.0				
positive conversation planning	leaving exploring	talk to higher rank travel / moving	travel with ship flee	battle goa'uld
				
Model 1: tfidf 0.01				
medical aid and injury health / medical stuff	commands in control room fight	problem solving, dangerous and powerful tech, moving back in time, war	contact with aliens commands from general Science fiction aliens @ new places	Flee, no time,danger goa'uld
				
Model 2: tfidf 0.02				
bad news, connected to past Traveling through Stargate, strategy	Travel worm holes, Sci-fi Science	saved from danger concerning galactic attacks (Sci-fi), War against Goa'uld	successfully following something Exploration & Strategy	Someone torn between host and parasite, emotional talk, Goa'uld
				

Model 3: tfidf 0.03				
sci-fi Travelling through & exploring space	communicate bad news History / people ??	get by a russian bodyguard Problems	scifi, unsafe worm holes Stargate?	emotional talk with father -
				
Model 4: tf-idf 0.04 (red/blue ratio 80%+)				
relief, got away war	phoning with doctor, injury fight, emergency	sci-fi war fight / flee	uncovering lies and mistakes travel & science ?	bad jokes, going in planning / people
				
Model 8: tfidf 0.08 (red/blue ratio 85%+))				
diplomacy, war strategy negotiation	fighting for freedom, by the grace of god war	finding valuable stuff fight on enemy ship	sci-fi injury Problems (science, injury)	war fights
				

C. Overview Generation of an Episode

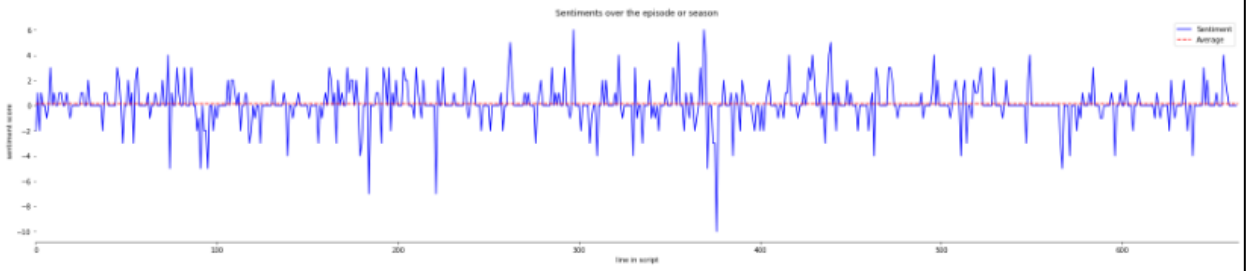
```
1 generate_overview_episode(1,1)
```

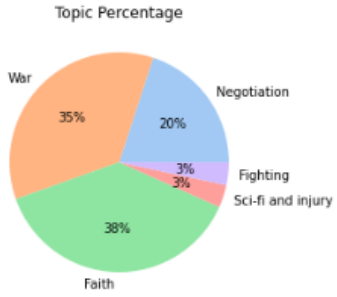
```
Shown is an overview over season  1  episode  1


There are 24 characters in the season.
These characters are airman, woman, officer, apophis, soldier, samantha_carter, jack_o_neill, hammond, warner, kawalsky, harrim
an, ferretti, daniel_jackson, skaara, sha're, teal_c, bo'la, boy, tech, medic, warren, native, goa'uld, and casey.


The 3 with the highest word counts are jack_o_neill, daniel_jackson, and samantha_carter.


In total, there are 6472 words spoken.
The sentiment score over the episode/season has an average of 0.1205.
The graph below shows the sentiment for each line in chronological order.
```





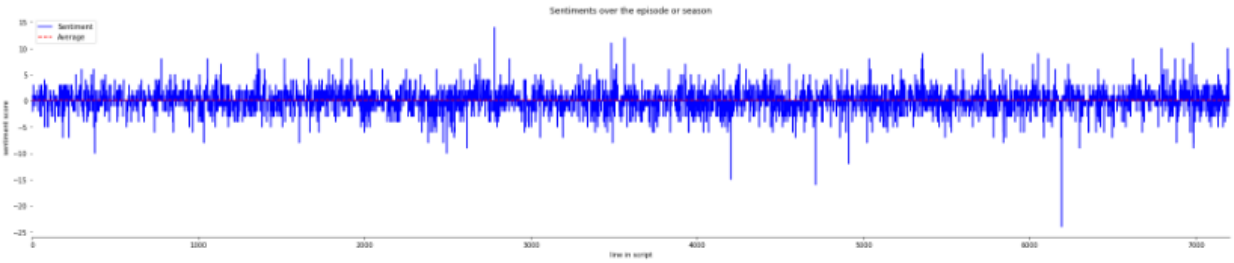
D. Overview Generation of a Series

Shown is an overview over season 1

There are 106 characters in the season.
These characters are airman, woman, officer, apophis, soldier, samantha_carter, jack_o_neill, hammond, warner, kawalsky, harriman, ferretti, daniel_jackson, skaara, sha're, teal_c, bo'la, boy, tech, medic, warren, native, goa'uld, casey, dr langford, scientist, catherine, martha, earnest, dr_janet, priest, bra'tac, o`neill, drey'auc, rya'c, nem, mackensie, kleinhouse, cole, hathor, solder, cassie, nurse, dr warner, davis, hanno, man, young hanno, villager, shak'l, narim, omoc, tuplo, maybourne, lya, silver, harlan, tv reporter, soldier 2, solder 3, airmen, jaffa, "auto destruct in, walter, doctor, kennedy, kinsey, entity, voice, ml, klorel, abu, ., mughal, turghan, nya, guard, makepeace, marine, leedora, johnson, connor, franks, baker, hanson, jamala, sara, dad, reporter, sara's dad, crystal, answering machine, police officer, charlie, secretary, nefrayu, oper, anteaus, alekos, thetyes, kynthia, argosian, gairwyn, thor, kendra, and unas.

The 3 with the highest word counts are jack_o_neill, daniel_jackson, and samantha_carter.

In total, there are 76413 words spoken.
The sentiment score over the episode/season has an average of 0.0303.
The graph below shows the sentiment for each line in chronological order.



The amount of speech each of the top 3 characters has per episode, is as follows: