Department of General Linguistics

Faculty of Arts

Palacký University Olomouc

Authorship Attribution of Fiction Based on Character Networks

Bachelor's Thesis

Author: Veronika Straková

Supervisor: Mgr. Vladimír Matlach, Ph.D.

Olomouc

2023

Declaration

I hereby declare that I have written this Bachelor's Thesis, "Authorship Attribution of Fiction Based on Character Networks", by myself under the supervision of Mgr. Vladimír Matlach, Ph.D., and that all the resources and literature are properly cited.

In Olomouc, June 24, 2023

....

Veronika Straková

Acknowledgements

I would like to express many thanks to Mgr. Vladimír Matlach, Ph.D. for his help, advice, guidance, and recommendations during the whole process of writing the thesis.

Abstrakt

Název práce: Určování autorství beletrie podle sítě postav

Autor práce: Veronika Straková

Vedoucí práce: Mgr. Vladimír Matlach, Ph.D.

Počet stran a znaků: 105, 133 072

Počet příloh: 10

Abstrakt:

Cílem této bakalářské práce je zjistit, zda kvantitativní vlastnosti sítí postav, které vytváří autoři beletrie, mohou přispět k určení autorství. Teoretická část práce se zabývá představením teorie sítí, teorie grafů a analýzy sociálních sítí. Dále jsou představeny sítě postav, následovány popisem procesu extrakce těchto sítí. Součástí práce bylo navrhnutí vlastního postupu pro extrakci sítí postav. Tento postup je v praktické části aplikován na 75 románů od pěti různých autorů, tj. výsledkem je 15 sítí postav pro každého autora. Na základě faktorové analýzy je vybrán pro další zkoumání vliv průměrného koeficientu shlukovaní, průměrné mezilehlosti a průměrné blízkosti sítí na určování autorství. Pomocí binární logistické regrese je zkoumáno pro každý pár autorů, zda je model s vybranými vlastnostmi signifikantní. Poslední část práce obsahuje zamyšlení nad tím, co lze z výsledků vyvodit, a jak výsledky ovlivňuje zvolená metodologie.

Klíčová slova: určování autorství, rozpoznávání pojmenovaných entit, sítě postav, faktorová analýza, binární logistická regrese, průměrný koeficient shlukování, průměrná mezilehlost, průměrná blízkost

Abstract

Title: Authorship Attribution of Fiction Based on Character Networks

Author: Veronika Straková

Supervisor: Mgr. Vladimír Matlach, Ph.D.

Number of pages and characters: 105; 133,072

Number of appendices: 10

Abstract:

The aim of this Bachelor's Thesis is to explore whether the quantitative features of character networks built by authors of fiction contribute to the authorship attribution task. The Theoretical Part of the thesis introduces network science, graph theory, and social network analysis. Then the character networks are introduced, followed by a description of the character network extraction process. As a part of this thesis, the author's own character network extraction process has been designed. It was applied during the Practical Part to 75 novels written by five distinct authors, i.e., 15 character networks per author were extracted. Based on factor analysis, the influence of average clustering coefficient, average betweenness centrality and average closeness centrality on the authorship attribution task has been selected for further study. Binary logistic regression has been used to assess for each pair of authors whether the model with the selected features is significant. The last part of the thesis includes a reflection about what can be inferred from the results, and a discussion about how the results are influenced by the methodological choices.

Keywords: authorship attribution, named entity recognition, character networks, factor analysis, binary logistic regression, average clustering coefficient, average betweenness centrality, average closeness centrality

Table of Contents

1 Introduction

A unique authorial style is what distinguishes one author from another. It might be manifested by the author's choice of lexemes and vocabulary richness, the length of the sentences or words, word frequencies, and many more features. By leveraging some of these features, the goal of the authorship attribution task is to identify the author of a given text (more in Stamatatos 2009). In our case, we are interested in the way an author builds the network of character interactions. Our aim is to explore whether we can attribute authorship based on the quantitative features of character networks.

A character network is a graph extracted from the narrative of a work of fiction. The characters are represented as nodes of the graph, and their interactions are represented as links between them. In order to gain a better understanding of character networks, the *Theoretical Part* offers an introduction to Network Science, Graph Theory and Social Network Analysis. Subsequently, character networks are introduced, and the process of their extraction is thoroughly described.

The execution of the *Practical* Part of the thesis consists of several sub-steps. First, the methodology behind data selection is presented, as well as the selected authors and novels that enter the analysis. Then, the approach we have adopted towards character network extraction is described. Subsequently the quantitative features of interest are extracted from the analysed networks. The correlation between the length of the novels and the network features is addressed. Based on factor analysis, three final features are selected for further study. Finally, for each pair of analysed authors, binary logistic regression is used in order to predict the authorship based on the selected network features.

The last part of the thesis presents a summary and a discussion of the obtained results, as well as a reflection about the qualitative assumptions that can be inferred from them. Lastly, we recommend a direction for future research, and we conclude whether selected quantitative networks features contribute to the authorship attribution task.

2 Theoretical Part

In the first three sections of the *Theoretical Part* of the thesis, we offer an introduction to Network Science, Graph Theory, and Social Network Analysis. In the last two sections, we present an introduction to character networks and a thorough description of the character network extraction process.

2.1 Network Science

Network science is an interdisciplinary field, quantitative, mathematical, and computational in nature, with focus on empirical data and utility. It profoundly impacts many sectors, such as economics, management, health, security, science etc. (Barabási, n.d.).

People encounter complex systems on everyday basis. Each complex system can be described as a network consisting of its components and the interactions between them. Barabási (n.d.) lists the following examples of complex networks:

- individual people who form the society, i.e., *social network*,
- cell phones, computers and satellites that form the communication infrastructures, i.e., *communication network*,
- generators and transmission lines that form the *power grid*,
- neurons that interconnect in the brain, i.e., *neural network*,
- genes, proteins and metabolites that interact in the cells of human body, i.e., *cellular network*,
- goods and services that form the *trade network*.

As evident from the listed examples, complex systems and the networks they form differ in form, size, shape and nature. However, most networks share the same organizing principles, in other words "despite the apparent differences, the emergence and evolution of different networks is driven by a common set of fundamental laws and reproducible mechanism" (Barabási 2012, 7).

In order to understand networks, one must first understand the Graph Theory upon which network science builds. We shall present some basics of Graph Theory in the following section.

2.2 Graph Theory

Whereas Barabási (n.d.) calls Network Science "the science of 21st century", the origin of Graph Theory dates back to 18th century to the mathematician Leonhard Euler who solved the Seven Bridges of Königsberg Problem by representing it as a graph (more in Tsvetovat and Kouznetsov (2011, 23-25) or Network Science by Barabási (n.d.)[1]).

Any network can be represented by the means of graph theory as a graph (i.e., a mathematical representation/simplification). A network consists of its components, called *nodes* or *vertices*, and the interactions between them, called *links* or *edges*[2] . The network has two prominent properties: *number of nodes* (N), i.e., size of the network; and *number of links* (L), i.e., total number of interactions between the nodes.

If the links interconnect nodes of one type of entity (e.g., people, organizations), we speak about a *1-mode network*; if the links interconnect nodes of two different types of entities (e.g., people & organizations; characters & chapters), we speak about a *2-mode (bipartite) network*. In the case of links that interconnect nodes of three and more types of entities, the network is *multimodal (multipartite*).

The links of a network can be either *directed* or *undirected*, depending on the nature of the interaction between the nodes of concern. If all of the network's links are directed, we speak about a *directed network*; if all its links are undirected, we speak about an *undirected network*. Some networks have links of both types. Furthermore, the links, and consequently the networks, can be either *weighted* or *unweighted*, depending on the nature of the interactions.

A graph can be "walked" through various algorithms. An algorithm walks in a predetermined order from the starting node through the edges to its neighbours, and then to the neighbours' neighbours etc. A *walk* is a sequence of nodes and edges that connect the nodes. The length of a walk can be measured – it is equal to the number of

[1] Albert-László Barabási, "Network Science by Albert-László Barabási," BarabásiLab, accessed June 8, 2023, http://networksciencebook.com/chapter/2#bridges.

[2] Barabási (n.d.) explains the difference between the two terminologies: the terms Network, Node and Link belong to the field of Network Science and refer to real systems; whereas the terms Graph, Vertex and Edge belong to the field of Graph Theory and refer to the mathematical representations of the networks. However, the terms are often used interchangeably, which is the case of this thesis too.

the edges walked through. A *path* is a walk that forbids the repetition of nodes. *Path length* is used to measure *distance* in networks.

Graph distance is an abstraction of a walk, and it can be measured in various ways. The *shortest path* is the simplest measure of distance. The shortest path between any two nodes is such a path that has the fewest links between the nodes. It can be called simply *distance* denoted by d_{ij} or d. There can be multiple shortest paths between a pair of nodes[3].

There are many other aspects of graph theory that could be covered in this section. However, for the purposes of this thesis, we shall content ourselves with the introduced basics, and refer the reader to the following sources: for introduction to the field, advanced topics, applications and illustrative examples, see Barabási (n.d.)[4], Tsvetovat and Kouznetsov (2011) or Zweig (2016).

2.3 Social Networks and Social Network Analysis

The aim of this thesis is to explore character networks, i.e., social networks built by the authors of fiction. Therefore, we will introduce the basics of Social Network Analysis (SNA) and its main concepts, such as *relationships* or *centrality measures*.

SNA might help with explanation and understanding of real-world social networks. Barabási (n.d.) and Tsvetovat and Kouznetsov (2011) list many examples of practical applications of SNA: it can help to uncover the power distribution within a company, or it can even be used to study the inner structure of terrorist organizations and cells.

The graph of our concern is a so-called social graph, i.e., a social network. A social network consists of a set of relationships. The actors involved in a relationship are represented in the graph theory by nodes, the relationship itself is represented by a link.

[3] There is a difference between undirected and directed networks to bear in mind: whereas in the case of an undirected network $d_{ij} = d_{ji}$ this is not necessarily the case of directed networks.

[4] Albert-László Barabási, "Network Science by Albert-László Barabási," BarabásiLab, accessed June 8, 2023, http://networksciencebook.com.

Relationships

Relationships are at the core of SNA. As Tsvetovat and Kouznetsov (2011, 2) state, in real world, it is challenging to determine what a relationship is and how to quantify its quality. One way to quantify a relationship is to consider the frequency of interaction, as the frequency is an objective and clear indication of emotional investment in a relationship.

A relationship can be either *binary* (there is/there is not a relationship) or *valued* (the relationship is assigned a weight based on some criteria, e.g., frequency), the former resulting into the extraction of unweighted, and the latter into the extraction of weighted networks. Furthermore, a relationship can be *symmetric* (e.g., friendship, romantic relationship) or *asymmetric* (e.g., employer & employee, teacher & student relationship). The links in some social networks are intrinsically asymmetric (e.g., a network representing who emails whom) and they result into the extraction of directed networks. In the opposite case, when all links are symmetric, the resulting network is undirected.

The goal of SNA is to analyse these relationships from the perspective of the whole network, and look for the answers to the following questions: *Who is important in the given social network? Who has the central position?* In order to answer these questions, several centrality measures can be calculated.

Centrality Measures

Answers to the above-mentioned questions will vary according to how importance, i.e., centrality, is perceived. The power and influence of nodes are measured by centrality measures. There are many centrality measures, for example Das, Samanta, and Pal (2018) list fourteen different types. However, in this thesis we shall focus only on four "classic" (Tsvetovat and Kouznetsov 2011, 45) centrality measures: 1. Degree Centrality, 2. Closeness Centrality, 3. Betweenness Centrality, and 4. Eigenvector centrality.

1. Degree centrality

Degree is a feature of each node that indicates the number of links it has to other nodes in the given network, i.e., it indicates the number of immediate neighbours of a given node. Degree is a feature of individual nodes; however, the whole network can be described by the *average degree* (*average degree centrality*).

Degree centrality finds the "local celebrities", i.e., nodes that are significantly more popular than others in the network (Tsvetovat and Kouznetsov 2011, 45). *Figure 1* represents the character network extracted from J. K. Rowling's *Harry Potter and the Philosopher's Stone*[5] where the size and the colour of a node reflect its degree. The network, as well as all the networks presented in this thesis, has been extracted using our character network extraction approach, described in section *3.3*. Not surprisingly, the character with the highest degree in the narrative is the main character, *Harry Potter*.

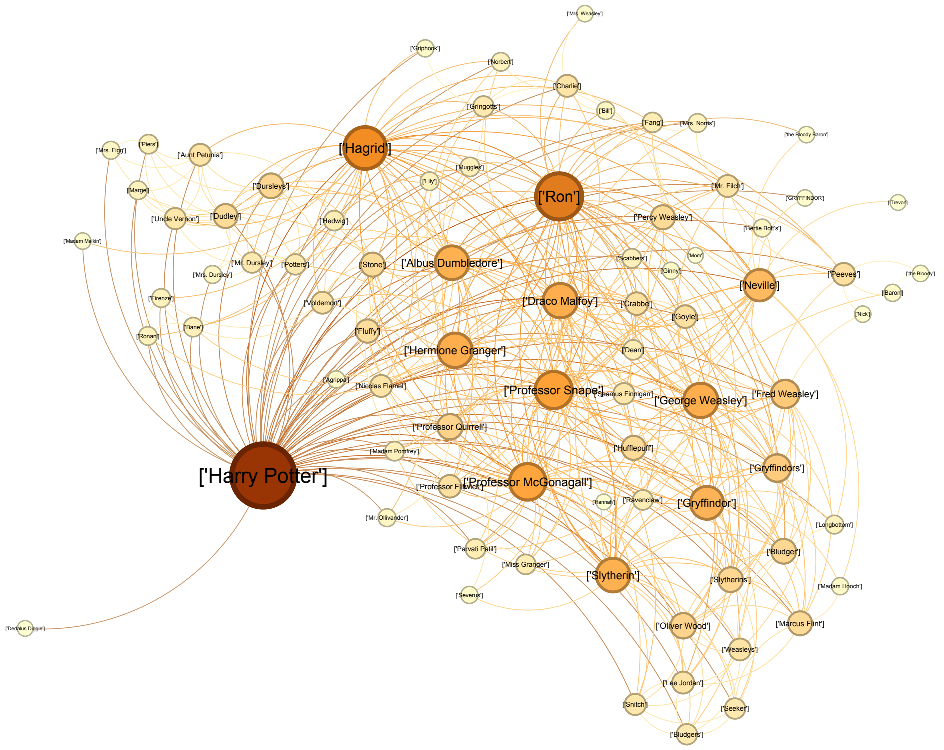


Figure 1: *Character network extracted from Harry Potter and the Philosopher's Stone. The size and the colour of the nodes reflect the degree centrality of the nodes. The bigger and the darker a node, the higher degree it has.*

[5] J. K. Rowling, *Harry Potter and the Philosopher's Stone* (Bloomsbury Publishing, 2014).

2. *Closeness Centrality*

Closeness centrality measures the distance, or inversely closeness, of a given node from the rest of the nodes in the network. For each node, it is calculated as the inverse of the average shortest path lengths between the node and all other nodes in the network[6] .

Nodes with high closeness centrality influence the information flow and they establish “a shared perception of the world” (Tsvetovat and Kouznetsov 2011, 50). *Figure 2* presents the same *Harry Potter* network as *Figure 1*, only this time the size and the colour of a node reflect its closeness. A lot of characters appear to have similar closeness, whereas only a few of them stand out. Most of the shortest paths lead through *Harry Potter* and are very short (the average path length is 2.039). It is not surprising, as most of the characters share the same spatiotemporal setting and know each other.

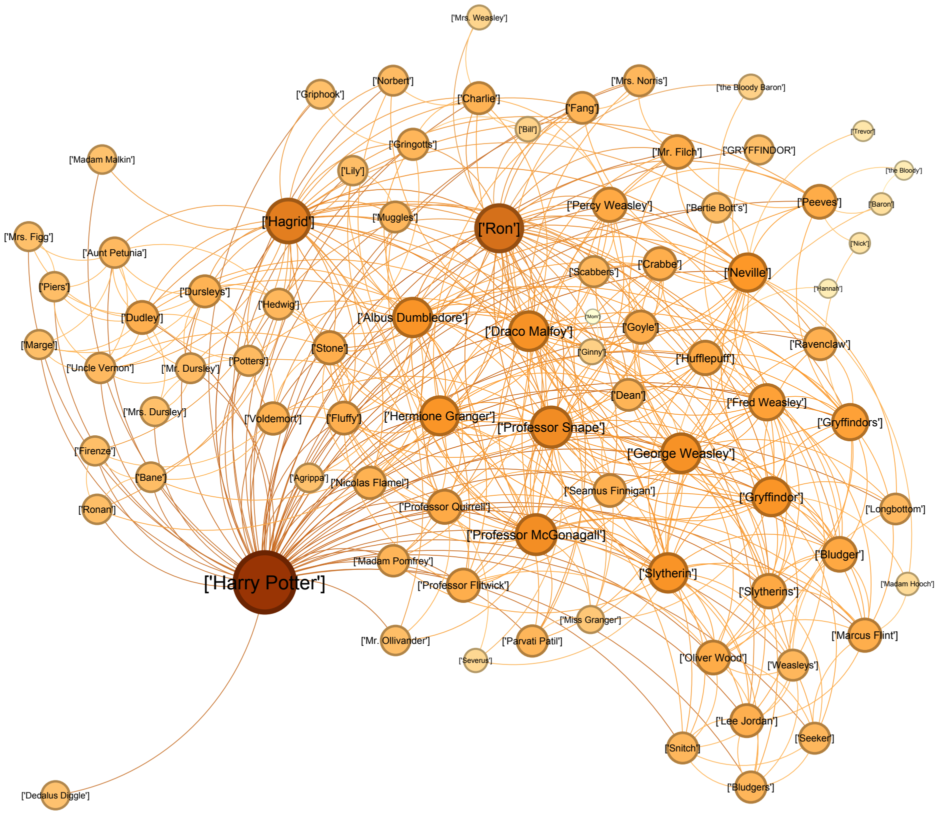


Figure 2: *Character network extracted from Harry Potter and the Philosopher's Stone. The size and the colour of the nodes reflect the closeness centrality of the nodes. The bigger and the darker a node, the higher closeness centrality it has.*

[6] This calculation leads to a normalized value of closeness centrality. Unnormalized closeness centrality is calculated as inverse of the sum of the shortest path lengths of a node.

3. *Betweenness Centrality*

Nodes with a high betweenness centrality are often so-called *bridges* or *boundary spanners* – they preside over a communication bottleneck, i.e., they often interconnect various communities/clusters of the graph (Tsvetovat and Kouznetsov 2011, 51). This position gives them considerable power over the information flow.

Betweenness centrality of a node measures how many shortest paths pass through it. The removal of a node with a high betweenness centrality would result into a disruption of the information flow. *Figure 3* shows the *Harry Potter* network where the size and the colour of a node indicate its betweenness centrality. A lot of shortest paths pass through the character *Harry Potter*. For instance, to the dismay of *the Dursleys*, *Harry Potter* links their ordinary family to the rest of the network, the wizarding world.

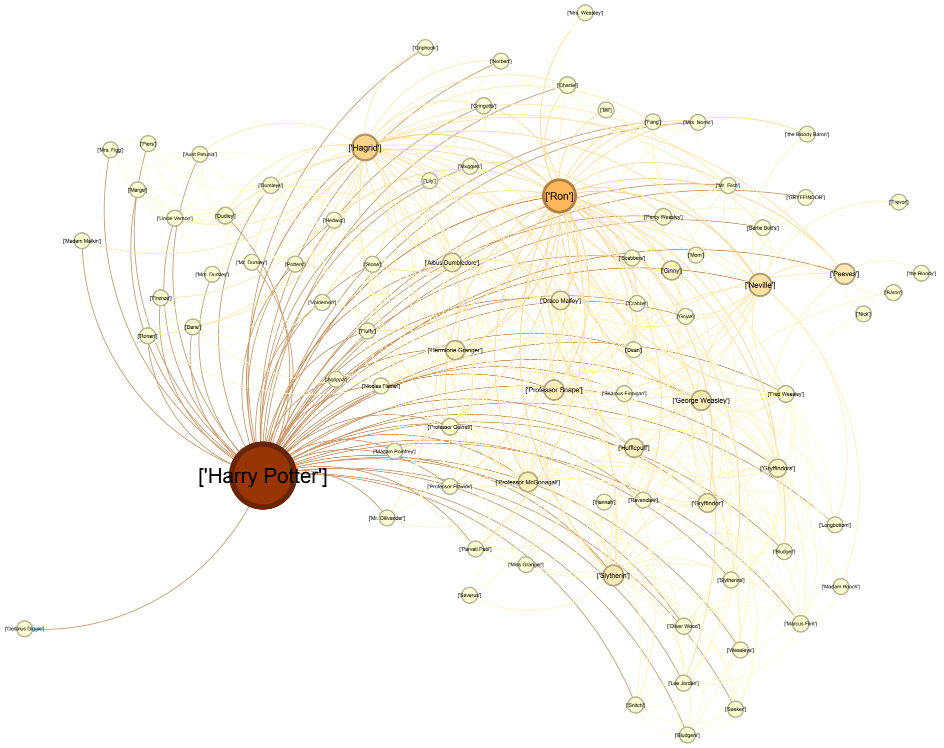


Figure 3: *Character network extracted from Harry Potter and the Philosopher's Stone. The size and the colour of the nodes reflect the betweenness centrality of the nodes. The bigger and the darker a node, the higher betweenness centrality it has.*

4. *Eigenvector Centrality*

A high value of eigenvector centrality indicates that the node in question has important, well-connected neighbours with "high scores", even though the node itself does not necessarily have a lot of connections. Connectedness to high-score nodes in turn increases the score of the given node.

The position of nodes with a high eigenvector centrality is a powerful one, as it grants access to the information, while simultaneously it allows to stay "largely in the shadows" (Tsvetovat and Kouznetsov 2011, 55). Using this measure, new important characters in the *Harry Potter* network emerge (see Figure 4), namely *Professor Snape* and *Professor McGonagall*. These characters are not the main ones, but they do influence the protagonists and steer their actions in a certain direction.

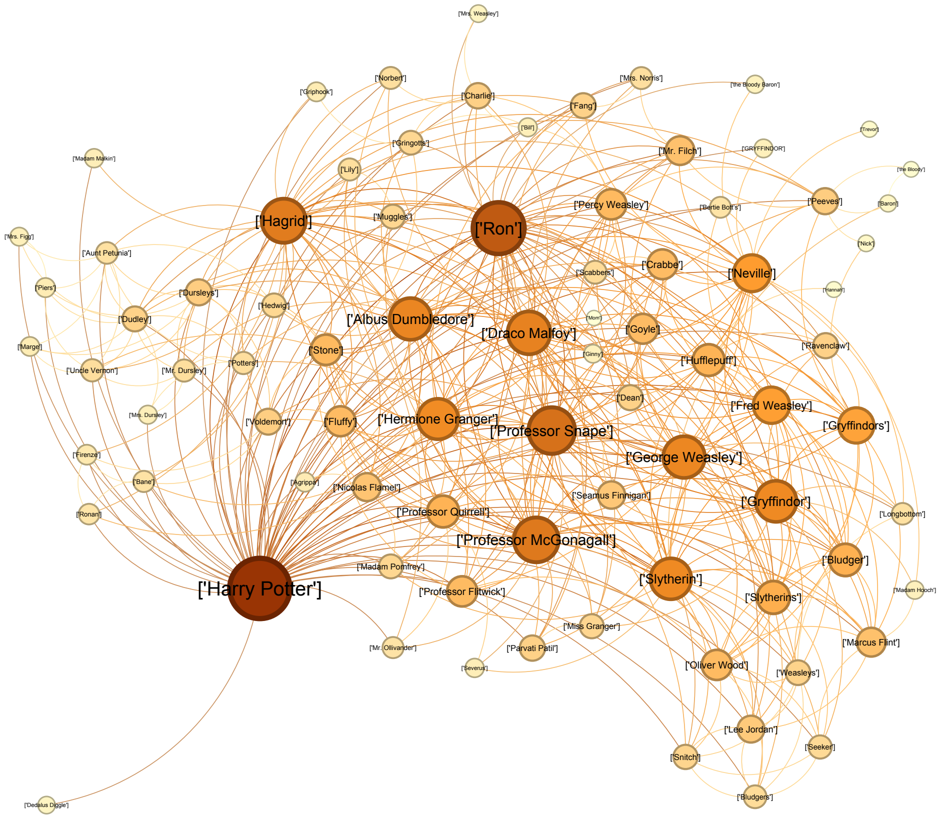


Figure 4: *Character network extracted from Harry Potter and the Philosopher's Stone. The size and the colour of the nodes reflect the eigenvector centrality of the nodes. The bigger and the darker a node, the higher eigenvector centrality it has.*

Other Network Features: Modularity and Clustering Coefficient

Modularity is a measure that indicates the strength of division of a network into modules (clusters, communities). If the modularity of a network is high, the links between the nodes within the modules are dense, and the links between different modules are sparse. The method used in this thesis for community detection is the Louvain Method (more in Blondel et al. 2008; or NetworkX documentation[7]).

Figure 5 shows the *Harry Potter* network divided into five communities; each community is represented by a different colour. For example, the orange community includes *Harry Potter* and the characters he interacts with *before* entering the wizarding world, such as his family members, *the Dursleys*, or his neighbour, *Mrs. Figg*.

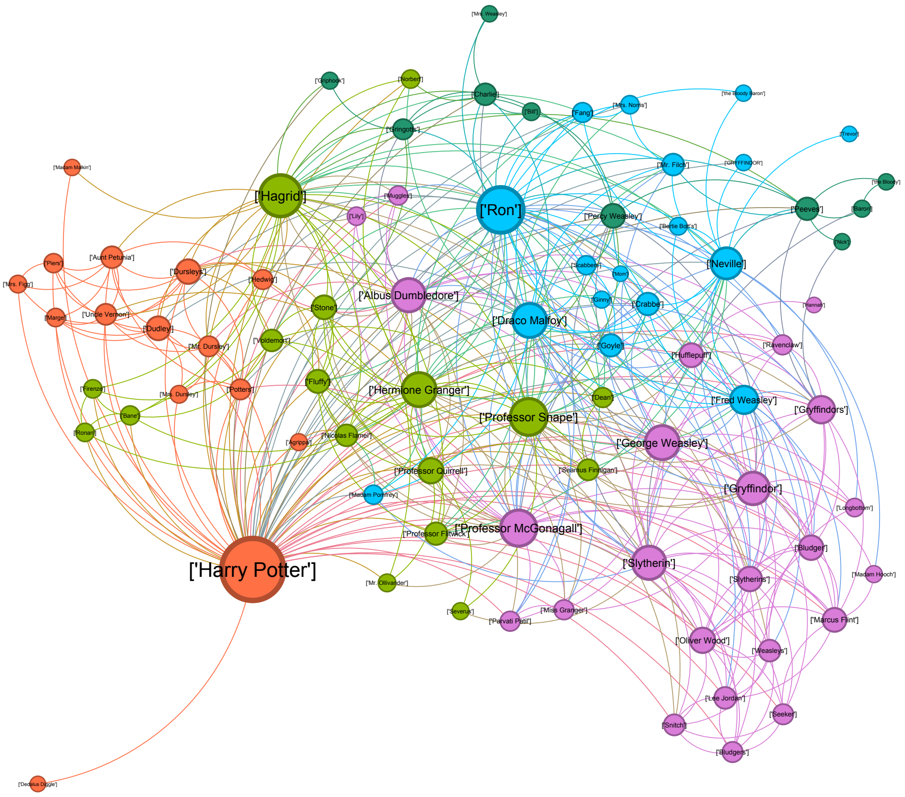


Figure 5: *Character network extracted from Harry Potter and the Philosopher's Stone. The size reflects the degree of the nodes. The colours represent distinct communities.*

[7] "Louvain_communities," NetworkX 3.1 Documentation, accessed June 8, 2023, https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.louvain.louvain_communities.html.

The *local clustering coefficient* takes the value between 0 and 1 and it indicates to what degree the neighbours of a given node link to each other. If the clustering coefficient of a given node is equal to zero, it indicates that *none* of its neighbours share a link among them; if the clustering coefficient of a given node is equal to 1, it indicates that its neighbours *all* share a link among them (more in Barabási, n.d.)

In other words, clustering coefficient measures local density of a network. *Figure 6* shows the *Harry Potter* network where the size and the colour of the nodes reflect their clustering coefficient. The figure offers quite a different picture to those presented earlier. The main character *Harry Potter*, usually prominent in the network, has a very low clustering coefficient due to a simple reason. It is a character with the highest degree, i.e., with the highest number of neighbours. Consequently, it is not surprising that not all its neighbours are interconnected.

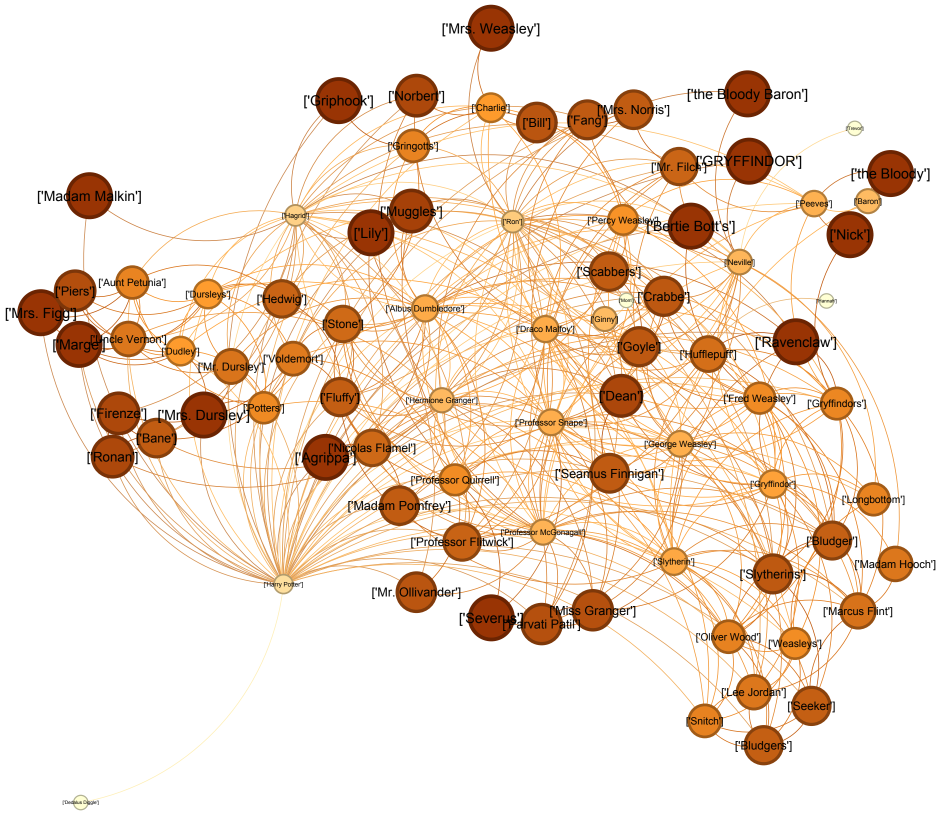


Figure 6: *Character network extracted from Harry Potter and the Philosopher's Stone. The size and colour of the nodes reflect their clustering coefficient. The bigger and darker a node, the higher its clustering coefficient.*

Clustering coefficient is a feature of individual nodes. However, the whole network can be described by the *average clustering coefficient*. Average clustering coefficient is the probability that two neighbours of a randomly selected node share a link.

There are many other features of social networks that could be explored in this section of the thesis. However, for the purposes of this thesis, the presented selection is sufficient.

2.4 Character Networks: An Introduction

In this section we discuss what a character network is and what we understand under the notion of work of fiction. Then we address the issue of typical features of fictional character names that make it difficult to extract character networks automatically.

2.4.1 What Is a Character Network?

A character network is a graph extracted from narrative. The vertices of the graph represent individual characters, the edges of a graph represent relationships (or interactions) between the characters. In their survey of character networks, Labatut and Bost (2019, 2) claim that "graphs are a natural modeling paradigm" because they enable the representation of a system and its study through the interaction of its constituting elements.

Labatut and Bost (2019, 2-3) comment on the application of character networks: through Character Network Analysis, one might obtain a simplification of the plot of a work of fiction, detect relevant patterns and events or identify a character's role. The automatization of the character network extraction process is used to solve different tasks, e.g., plot summarization, genre classification, role detection etc. The aim of this thesis is to explore whether character networks as built by the author might contribute to the authorship attribution task.

2.4.2 A Work of Fiction

Fiction is a creative work which usually takes the form of a narrative, i.e., an account of a series of related events or experiences. Such a work habitually features imaginary individuals (characters), events and places, and does not follow real-world history or facts. In the broad sense, fiction refers to any imaginary narratives regardless of the medium (e.g., dramas, films, series, comics etc.) In the narrow sense, fiction refers to written narratives in prose (e.g., novels, short stories, fairy tales etc.). In this thesis, we shall understand fiction in its narrow sense – we are interested solely in character networks extracted from novels.

2.4.3 Features of Fictional Character Names

In order to extract a character network from fiction, the first step is to identify the characters in the text. That is not always an easy task, especially for a machine, as the fictional character names have specific features. Labatut and Bost (2019, 4) point out some of these, and we supply examples from J. K. Rowling's *Harry Potter* series.

Fictional characters are often members of the same family and thus they share family name (e.g., *Mr.* and *Mrs. Weasley, Ginny, Ron, Fred, George, Percy, Charlie, Bill Weasley)*; they bear nicknames or are addressed by hypocorisms (e.g., *Moody vs Mad-Eye, Ronald vs Ron*), their names are preceded by specific honorifics (e.g., *Auror Alastor Moody*), or their names convey certain meanings or functions (e.g., *Luna "Loony" Lovegood*). It is not uncommon that characters are non-anthropomorphic beings or inanimate objects (e.g., *Sorting Hat*).

The writers can choose from a multitude of aliases to refer to their characters. Humans performing the character identification task will leverage the semantic content of the work of fiction, their extralinguistic knowledge and the knowledge of situational and communicative context. However, the task is a challenging problem for a machine (Labatut and Bost 2019, 4): a partially automated approach towards character detection will depend on an external database of names (a *gazetteer*); a fully automated approach will depend on the corpus the model had been trained on. Given the unique characteristics of fictional character names, the model might be presented by a situation it had never encountered before.

2.5 Character Network Extraction Process

Here, we bring a detailed overview of the steps of the character network extraction process based on the survey by Labatut and Bost (2019). They present a process of character network extraction that consist of the following three steps: 1. Identification of Characters, 2. Identification of Characters' Interactions, and 3. Graph Extraction.

The first step of the first stage is to detect occurrences of the characters in the work of fiction and decide whether the character unification task will be performed or not. The output of the first stage is a chronological sequence of (unified) character occurrences.

The second stage consists of detecting interactions between the characters. The output of this stage is a chronological sequence of interactions between characters.

The last stage relies on the list acquired from the first two stages. Before constructing the network itself, one might simplify the list by filtering/merging some of the characters. The last step is to decide whether the network is going to be static or dynamic.

2.5.1 Identification of Characters

The goal of the character identification task is to answer the following question: *what* characters appear in the narrative and *when* exactly they appear (Labatut and Bost 2019, 6). To identify characters in novels is not an easy task. On the contrary, Elson (2012, 20) states: "Character identification in novels is made complicated by the fact that there are myriad of ways in which authors refer to characters." He offers the following possibilities of character references: a named entity (e.g., *Harry Potter*), aliases and variations (e.g., *Mr. Potter*), pronouns (e.g., *he*), and descriptive nominals (e.g., *the student, the old man*). During the unification step, mentions that co-refer to the same character are unified.

Labatut and Bost (2019) distinguish between manual approaches, partially automated and automated approaches toward character identification.

A. Manual Approaches

One of the possibilities is to rely on a fully manual approach, for example through direct annotation (manual annotation of the narrative), as Agarwal et al. (2012) did for Lewis Carrol's *Alice in Wonderland*; or character indices (predefined resources constituted manually for certain classic fictions containing information about occurrences), as Rochat and Kaplan (2014) did for Rousseau's *Les Confessions*. One might choose this approach due to technical limitations, but it might be also a methodological choice. The advantage of manual approach is that character occurrence detection and character occurrence unification can be performed simultaneously. On the other hand, manual annotation of numerous novels would require significant amount of time and energy.

B. Partially Automated and Automated Character Identification

When dealing with partially automated and automated approaches towards character identification, the task is divided into two parts. Firstly, character occurrences in the considered novel must be detected. Secondly, it must be decided whether character occurrences unification will be performed and if yes, how to proceed.

1. Detection of character occurrences

As stated earlier, the characters might be referred to in many ways. According to Labatut and Bost (2019, 8), all methods used in the surveyed literature address the detection of occurrences of proper nouns/named entities. However, some authors do not handle the detection of pronouns and nominals because this task is considered more complex and does not necessarily bring much relevant information.

The first partially automated method is to use *a predefined list of names ("a gazetteer") and proceed through exact matching*. These *gazetteers* are constituted manually by the researchers or through an external source (e.g., Wikipedia page containing all the characters from the considered novel). The constitution of such a list is challenging, due to the number of possible references. Grener et al. (2017) use this approach to generate dynamic networks of Dickens' novels.

The second method is to approach the character occurrences detection task as a specific type of *Named Entity Recognition (NER)* task. NER is one of the tasks of Information Extraction and automated Natural Language Processing (NLP). The aim of a NER system is to identify named entities in unstructured free text and classify them into predefined categories. Named entities are expressions in the text referring to the names of people, locations, organizations, works of art, but also time and quantitative references (more in Nadeau and Sekine 2007).

According to Labatut and Bost (2019, 8) a commonly used method to detect character occurrences is to apply an off-the-shelf NER tool and disregard all the entities except the person entity. As novels have specific features that cause problems when using automated methods, some (optional) post-processing can be used after the application of the NER tool. For example, Sack (2012) removes proper names occurring less than five times and verifies the output manually. Ardanuy and Sporleder (2014) use a predefined list of honorifics to detect these in the text and to check the following text unit for the presence of a proper name; Trovati and Brady (2014) look for groups of words indicating possession through the presence of genitive case.

Labatut and Bost (2019, 8) note that performing these additional actions increases the probability of detecting non-human characters with human characteristics in terms of behaviour, as well as nominals. Depending on the language, one should also consider the detection of pronouns. For English, it is efficient to proceed through exact matching based on a manually constituted list, as done for example by Elson (2012).

2. *Character Occurrences Unification*

As mentioned above, every single character might be referred to by their full name (e.g., *Ronald Weasley*), any number of variations and aliases (e.g., *Ron, Mr. Weasley, Weasley*), pronouns (*he*) and nominals (e.g., *student, brother*). The goal of the character occurrences unification is to determine the referent of each detected occurrence.

According to Labatut and Bost (2019, 11), this step is often omitted due to two main reasons: this task is more difficult to perform than mere character detection, and sometimes it is simply not necessary. Dekker, Kuhn, and Van Erp (2019, 16) point out that sometimes it is even desirable to retain various aliases, as they bring new information. They show it on the example of *A Game of Thrones* character *Daenerys Targaryen*. Whereas the character is called *Dany* within friendly environment, she is addressed as *Daenerys Targaryen* by her enemies in her absence. Van Dalen-Oskam (2005) argues that the ratio between first name and last name occurrences reflects the level of intimacy in novels (quoted in Van Dalen-Oskam et al. 2014, 122).

The task of identifying sequences of expressions that represent the same concept is called *Coreference Resolution*. There are generic tools that are designed to solve the coreference resolution problem, but according to Labatut and Bost (2019, 11) their application to the works of fiction can be problematic.

Labatut and Bost (2019, 11) state that most authors use some form of name clustering where each cluster corresponds to all the names detected for a certain character. To do so, they take advantage of two factors: firstly, *string similarity*, and secondly, *gender compatibility*. The gender can be assigned based on gendered honorifics, such as *Mr*. or *Mrs*., and on gendered first names, such as *John* or *Jane*, matched against a manually constituted list or an external source. Some authors perform direct comparisons through predefined patterns and rules (see Ardanuy and Sporleder 2014; Vala et al. 2015).

Labatut and Bost (2019, 12) state that the resolution of other types of referents is an even more challenging task. For instance, the actual referent of a pronoun or a nominal might not appear at all during the whole course of a novel; or their referent might be split (e.g., *they, the Weasley brothers* can refer to all the male *Weasley* characters, except the father).

2.5.2 Identification of Characters' Interactions

The second step in the extraction process is to detect the interactions that occur in the narrative between each pair of the characters. There are several ways of defining what an *interaction* is. Labatut and Bost (2019) have identified five different approaches in the literature, based on 1. Co-occurrences, 2. Conversations, 3. Mentions, 4. Direct Actions and 5. Affiliations. In this section we will describe the approaches and supply each with an example from our illustrative novel, J. S. Fletcher's *Scarhaven Keep*[8].

1. *Co-occurrences*

According to the survey of Labatut and Bost (2019, 14), the interaction detection based on co-occurrences is the most frequent approach used in the literature. The relationship between characters is based on their joint appearance in the same unit of text. Firstly, the work of fiction must be divided into smaller *narrative units*. If two characters appear in the same narrative unit, a relation is created between them.

This approach is considered the easiest one, nevertheless it entails some limitations. It is not a very precise way of defining interactions; co-occurrence is "only a proxy for actual interaction" (Labatut and Bost 2019, 14): two characters may appear together but not interact together. The co-occurrence-based approach in theory

[8] J. S. Fletcher, *Scarhaven Keep* (Urbana, Illinois: Project Gutenberg, 2006).

also contains the other approaches within itself. This makes it impossible to assign the interactions a direction, and such interactions are considered bilateral.

The co-occurrence-based approach depends on the chosen narrative unit. Different authors choose different narrative units, ranging from several words (Hutchinson, Datla, and Louwerse 2012), 1 sentence (Lee and Yeung 2012), 1 paragraph (Elsner 2012) to 10 paragraphs (Elson, McKeown, and Dames 2010), a page (Rochat and Kaplan 2014) or a chapter (Chen and Wang 2016). Labatut and Bost (2019, 14-15) point out the limitations of different choices: a segmentation based on physical aspects (word spans, a page) is arbitrary, therefore likely to split complete units of text and miss co-occurrences; a segmentation based on sentences, paragraphs or chapters does not take into account that their length may vary considerably from one author to another.

Example (1) shows co-occurrences-based interaction pairs extracted from J. S. Fletcher's *Scarhaven Keep*, where the narrative unit is one sentence.

(1) *Stafford* was back at Scarhaven before breakfast time next morning, bringing with him a roll of copies of the Norcaster Daily Chronicle, one of which he immediately displayed to *Copplestone* and *Mrs. Wooler*, who met him at the inn door. (chapter 7)

Detected Co-occurrences: (Stafford, Copplestone),
(Stafford, Mrs. Wooler),
(Copplestone, Mrs. Wooler)

2. *Conversations*

Another approach to interaction detection is to consider characters' explicit verbal interactions and extract a *conversational network*. A conversation is asymmetric by nature, which makes it possible to extract a directed network. Conversational networks consider only the utterances in the form of direct speech. Their detection is not an easy task as it consists of many sub steps: quote detection, quote attribution and addressee identification. The difficulty of the automated process is why some authors (e.g., Moretti 2011) extract the conversation pairs manually.

According to Ardanuy and Sporleder (2015, 12) plays are the most appropriate form of a work of fiction where the conversation-based approach applies, as they are written in forms of a dialogue, whereas in novels a lot of action is not part of any conversation.

Example (2) shows a conversation-based interaction pair extracted from J. S. Fletcher's *Scarhaven Keep*.

(2) "Do you think that will do much good?" asked *Copplestone*.
"It depends upon the amount," replied *Mrs. Greyle*. (chapter 7)

Detected Interaction: (Copplestone, Mrs. Greyle)

3. Mentions

Another type of approach is based on explicit mentions of characters during conversations. An interaction edge is created if one character speaks about another (instead of speaking to). Similar to the previous approach, quote detection and quote attribution must be performed, however the addressee is of no concern. According to Labatut and Bost (2019, 20), in most cases the direction is assigned to the interactions, as mentions are intrinsically asymmetric.

Example (3) shows a conversation-based interaction pair extracted from J. S. Fletcher's *Scarhaven Keep*.

(3) "I'll tell you how it was," said *Mrs. Salmon*, seating herself and showing signs of a disposition to confidence. "*Miss Chatfield*, she'd been here, I think, three days that time--I'd had her once before a year or two previous…" (chapter 18)

Detected Interaction: (Mrs. Salmon, Miss Chatfield)

4. Direct Actions

This approach considers all types of direct actions that one character can perform on another (e.g., recalling someone), or that two characters can perform together (e.g., kissing). This approach allows to detect interactions that are not a part of a verbal action, and therefore it is well suited for the interaction detection in novels. It is however a complicated task, as it involves identification of the action and identification of the characters performing it and undergoing it. Depending on the type of action, the interaction is either naturally unilateral or bilateral. Some authors include only certain types of actions: for example, Bossaert and Meidert (2013) focus on the act of support among students in the *Harry Potter* series.

Example (4) shows a direct-action-based interaction pair extracted from J. S. Fletcher's *Scarhaven Keep*.

(4) *Copplestone* suddenly laughed and touched *Sir Cresswell* 's arm. (chapter 12)

Detected Interaction: (Copllestone, Sir Cresswell)

5. Affiliations

The affiliation-based approach defines interaction in terms of different states rather than actions. For example, this approach would create a link between family members, married couples, or members of the same social groups. According to Labatut and Bost (2019, 21) this approach is an infrequent one.

Example (5) shows an affiliation-based interaction pair from J. S. Fletcher's *Scarhaven Keep*.

(5) *...Mr. Bassett Oliver* is the younger brother of *Rear-Admiral Sir Cresswell Oliver*... (chapter 5)

Detected Interaction: (Mr. Bassett Oliver, Rear-Admiral Sir Creswell Oliver)

2.5.3 Graph Extraction

Labatut and Bost (2019) guide us also through the last stage of character network extraction: the extraction of the graph proper. In this stage, vertices and edges must be defined. The resulting graphs can be undirected or directed; unweighted or weighted; unsigned or signed. The temporal integration also must be decided.

Vertices usually represent individual characters, but they might also possibly represent a group of characters.

Regarding the *edges*, more aspects of the interaction must be defined, namely interaction *laterality, score, polarity* and *temporality*. Unilateral interactions are usually represented by directed edges, bilateral interactions can be either represented by undirected edges or by pairs of reciprocal directed edges. The interaction intensity is measured by the score, and it is represented by edge weights. If the score is signed, the edge weights can be either positive or negative according to the polarity of the interaction. Depending on how the temporality of the graph is handled, the resulting graph is either a *static graph* or a *dynamic graph*.

1. *Static Networks*

According to the survey of Labatut and Bost (2019, 22), the literature presents mostly static character networks. A static network is such a network that considers the complete set of interactions between the characters over the whole narrative, i.e., it is a *complete temporal integration*. This “bigger picture” representation of interactions and time might lead to omitting important details and hence significant information loss.

Deriving an edge from a sequence of interactions usually depends on the existence of the interaction or its frequency. Interaction aggregation is the simplest type of temporal integration. An edge is created between a pair of characters, if the total number of their interactions across the whole narrative is at least one; resulting network is unweighted.

2. *Dynamic Networks*

For some of the applications of character network analysis, the chronology of interactions and their development is crucial (Labatut and Bost 2019, 24). A dynamic network presents a sequence of character graphs (so called “time slices” by Labatut and Bost) where each graph represents a narrative unit of the novel. In other words, a dynamic network consists of a larger number of static networks. Temporal integration of interactions is still present, but this time it is performed over a much shorter period of time.

According to Labatut and Bost (2019, 24-25), the most widespread approach to extract dynamic networks is to use a fixed-sized temporal window. For novels, this window usually corresponds to one chapter, although e.g., Seo et al. (2014) divides the novel into 10 disjoint pieces. Labatut and Bost (2019, 25) also mention a special type of dynamic network: a cumulative network, also called an “incremental network” by Waumans, Nicodème, and Bersini (2015). Such a network entails all the interactions starting from the very beginning of the novel and ending at the considered time of the novel.

3 Practical Part

In the following section, we describe the approach we have adopted in order to conduct the practical part of the thesis. We analyse 75 novels written by five distinct authors based on three selected quantitative network features. For each pair of authors (i.e., 10 tests in total) we use logistic regression to test whether the features contribute to authorship attribution task.

3.1 Data Selection

In order to attempt to reach general conclusions about authorship attribution based on the quantitative features of character networks, we need to analyse a large quantity of novels written by numerous authors. The sample size is chosen in line with the goal and the scope of the thesis. We settle for 75 novels written by 5 distinct authors, i.e., 15 novels per author. It is reasonable to expect 15 novels written by an author, whereas acquiring a larger sample per author would be quite challenging.

The selected sample size will allow us to examine the influence of three distinct network features on the authorship attribution task (we discuss the number of variables in the section *4.6.1 Selection of Independent Variables*).

Additional Constraints on Data Selection

As we focus solely on the classification of authorship, we want to eliminate as much as possible results that could be attributed to other aspects of fiction than authorial styles. For example, we want the quantitative network features to be representative of the authors of the novels, and not of different genres. Therefore, it is desirable for our data to be as uniform in terms of genre as possible.

Another factor to consider is the varying length of the novels. Ideally, all selected novels would be of the same length, as we make the assumption that the network features correlate with the length of the novels.

The last constraint we apply is that the novels selected for the analysis must be 3rd person narratives, due to a simple reason: our approach to character network extraction does not consider pronoun coreference resolution, which in turn could lead to omitting many character occurrences of the 1st person narrator.

3.2 Our Data: Analysed Authors & Novels

Here we present the final data selected for the analysis. We analyse 75 novels written by 5 distinct authors, available at Project Gutenberg[9]. We consider the works of fiction written by the following authors:

1. Alger, Horatio, Jr. (1832 – 1899), for the most part of this thesis referred to as *Author_1*,
2. Altsheler, Joseph Alexander (1862 – 1919), for the most part of this thesis referred to as *Author_2*,
3. Ellis, Edward, Sylvester (1840 – 1916), for the most part of this thesis referred to as *Author_3*,
4. Henty, George Alfred (1832 – 1902), for the most part of this thesis referred to as *Author_4*,
5. Optic, Oliver (1822 – 1897), for the most part of this thesis referred to as *Author_5*,

for the list of individual novels with the indication of their length in tokens, see *Appendix A*.

All selected novels are classified as Children's Fiction / Juvenile Fiction; some of them with features of historical or adventure fiction, occasionally with some didactic features. We consider that we have met two of the three additional criteria presented above: all novels are 3rd person narratives, and all belong to the same genre. However, we have not been able to acquire 75 novels of the same length. We address the issue of possible correlation between the length of the novels and the network features in section *4.5. Are Network Features and the Length of the Novel Correlated?*

Pre-processing

Before applying the character network extraction process to the studied novels, all instances of text that do not form part of the narrative, such as preface, author's biography, author's note, contents etc., are removed from the data, so that each analysed novel starts with the first chapter and ends with the end of the narrative.

[9] Project Gutenberg is an online library of free eBooks, more information at https://www.gutenberg.org/, accessed April 3, 2023.

3.3 Character Network Extraction Process: Our Approach

The following pages describe in detail the approach we have adopted towards the extraction of character networks from novels. Whenever possible, we supply examples from our illustrative novel, *Scarhaven Keep* by J. S. Fletcher.

First, we have discarded all manual approaches, as they are not compatible with our aim. We analyse 75 novels, and to annotate characters manually would be enormously time-consuming. Therefore, we consider partially automated and automated approaches towards character network extraction. We work in programming language *Python*[10], and we supply the source code in *Appendix B.*

We have decided to make a slight change to the order of stages of extraction process as described by Labatut and Bost (2019). We divide our extraction process in four stages: 1. Character Occurrences Detection, 2. Identification of Character Occurrences' Interactions, 3. Unification of Interacting Characters, 4. Graph Extraction. We have changed the order of Character Unification and Interaction Detection, as with our coding skills it is easier to identify the interactions first and to unify the interacting occurrences second. We think that the order of performing these two tasks is arbitrary and does not result in different output.

3.3.1 Character Occurrences Detection

The goal of the first stage of our approach is not to identify characters but only to detect occurrences. To use gazetteers and proceed through exact matching is a straightforward way of character occurrences detection, nevertheless manual constitution of such a predefined list of characters for the sample novels would be unproportionally time-consuming. To extract gazetteers from external sources is also not possible because there are not any external lists of characters for the sample novels.

To use an off-the-shelf NER tool with optional post-processing is the most promising and appropriate method for our use.

[10] "Welcome to Python.Org," Python, accessed June 7, 2023, https://www.python.org/.

Off-the-shelf NER tool

We work in Python, and we have decided to use spaCy[11] for Natural Language Processing (NLP) NER task. SpaCy is the right choice for us, as it is user-friendly and suitable for users not extensively familiar with NLP, it is designed "to get things done" (spaCy, n.d.-b).

SpaCy is a free, open-source library for advanced Natural Language Processing in Python. It is designed specifically for production use and it "keeps the menu small" (spaCy, n.d.-b), i.e., it allows the user to easily handle tokenization, POS-tagging, lemmatization etc. More interestingly, it also features Named Entity Recognition:

> spaCy can recognize various types of named entities in a document, by asking the model for a prediction. Because models are statistical and strongly depend on the examples they were trained on, this doesn't always work *perfectly* and might need some tuning later, depending on your use case. (spaCy, n.d.-b)

spaCy pipeline

The trained model and pipeline we use is *en_core_web_trf*, i.e., transformer-based English pipeline trained on written web text, which features current-state-of-art NER.

1. ***Modification of default en_core_web_trf pipeline***

Before loading the spaCy *en_core_web_trf* model on the studied novels, we modify the pipeline. The resulting pipeline is from now on called *modified_default_pipeline*. We take two aspects into consideration: 1. Removal of *non-person* entities, 2. Expansion of *person* entities.

The default *en_core_web_trf* pipeline returns all named entities it encounters, such as *person, location, organization* etc. As we are dealing with character networks, we are interested only in *person* entities; therefore we remove all *non-person* entities. The default *en_core_web_trf* pipeline also ignores the potential presence of titles or honorifics (e.g., *Mr., Mrs., Professor*). The detection of such a title or honorific might be crucial for correct character identification during the unification stage (e.g., *Dursley* vs. *Mr. Dursley* vs. *Mrs. Dursley*).

[11] "SpaCy · Industrial-Strength Natural Language Processing in Python," spaCy, accessed June 8, 2023, https://spacy.io/.

We modify the pipeline to "expand" the named entities, following the spaCy guide (spaCy, n.d.-a). We use a rule-based approach to check for each encountered entity whether the token preceding the entity belongs to a predefined list of titles and honorifics ("*titles_and_honorifics_list*"). This list contains 1,238 items, and it is a variation of the *python nameparser titles configuration constants*[12] modified according to the grammar and punctuation rules of English (see GrammarBook.com 2021; University of Sussex, n.d.)[13]. To see the list, go to *Appendix C*.

Example (6) show expanded *person* entities detected in J. S. Fletcher's *Scarhaven Keep*.

(6) You think it wise?" asked *Sir Cresswell*. (chapter 12)
"Thank you, *Captain Andrius*," she said coolly. (chapter 20)

2. Evaluation of modified_default_pipeline

In this section, we evaluate the performance of the *modified_default_pipeline* (i.e., the pipeline modified to remove *non-person* entities; and to expand *person* entities). Nadeau and Sekine (2007, 12-15) describe the evaluation process in their survey: the output of a NER system is usually compared to the output of human linguists; and the system might produce different type of errors (e.g., missed entity, incorrectly labelled entity, partially-correctly detected entity etc.).

To report the accuracy of a NER system, *F-score* is used. *F-score* is the harmonic mean of *precision* and *recall*. *Precision* is defined as the number of relevant retrieved elements (i.e., true positives) divided by all retrieved elements (i.e., true positives & false positives). *Recall* is the number of relevant retrieved elements (i.e., true positives) divided by the number of all elements that should have been retrieved (i.e., true positives & false negatives).

[12] "HumanName Class Documentation," Nameparser 1.1.2 Documentation, accessed April 4, 2023, https://nameparser.readthedocs.io/en/latest/modules.html#module-nameparser.parser.

[13] *Python nameparser titles configuration constants* include 619 titles and honorifics, in lower case without further interpunction (e.g., *mrs, dr, doctor*). To meet the rules of English grammar and punctuation, we generate our *titles_and_honorifics_list* that consists of *python nameparser titles configuration constants* in capitalized form and of *python nameparser titles configuration constants* in capitalized form followed by a full stop.

To evaluate the accuracy of the *modified_default_pipeline*, we have used J. S. Fletcher's *Scarhaven Keep* as the sample text. We have manually annotated the characters occurring in Chapter II of the novel (3,784 tokens) and then we have loaded the pipeline on the sample text. We evaluate the pipeline's ability to find the correct type of entity (we are interested solely in the entity *person*). We consider the pipeline to be successful if it identifies an entity of the type *person* as a *person* entity; we do not consider whether the entity has been detected in its entirety (i.e., if a multiword entity, such as *Mary Wooler* is detected as a single-word entity, such as *Mary* or *Wooler*, we consider the detection to be successful). The results of the evaluation of *modified_default_pipeline* can be seen in *Table 1*.

	modified_default_pipeline
Precision	0.955
Recall	0.914
F1-score	0.934

Table 1: *Evaluation of modified_default_pipeline.*

3. Further post-processing of modified_default_pipeline

To improve the performance of the *modified_default_pipeline*, we take three additional steps that consist of: I. Discarding infrequent entities, II. Generating aliases for entities occurring at least three times, III. Implementing final measures.

I. Discarding infrequent entities

The output of *modified_default_pipeline* renders also false positives. Therefore, we have decided to discard occurrences detected less than 3 times. Below we present our reasoning for doing so.

First, if an occurrence is detected only once, it is likely an error (see *Figure 7* for an example from *Scarhaven Keep*: it might be an incorrectly classified entity, e.g. *GREY ROCK*; an incorrectly spelled entity, e.g., *Cobblestone*, an entity merged with interpunction marks and/or grammatical constructs, e.g., *Copplestone!—keep*, or an instance of entity that is not desirable to detect, such as an entity in possessive case, e.g., *Zachary Spurge's*; or a named entity in upper-case, e.g., *ELKIN*).

```
'Mr. Rothwell': 1, 'GREY ROCK': 1, 'GREY SEA': 1, 'Rutherford': 1, 'Mary
Wooler': 1, 'Basset Oliver': 1, 'the Sugar-Loaf': 1, 'Haskett': 1,
'Copplestone!--keep': 1, 'Mr. Stephen Greyle': 1, 'Mr. Marston': 1,
'Norcaster': 1, 'Lady Hartletop': 1, "Peeping Peter's": 1, 'Copplestone!--
every': 1, 'Marston Greyle!': 1, "Mrs. Greyle's": 1, 'Woolsack': 1,
'Master Chatfield': 1, "Zachary Spurge's": 1, 'Marris': 1, "Zachary
Spurge'll": 1, 'Ewbanks': 1, 'Mr. Greyle--': 1, 'Mr. Foreman': 1,
'DENNIE': 1, 'Thespis': 1, 'Dennie': 1, 'Gaines': 1, 'The Marston Greyle':
1, 'Prince Rupert': 1, 'Marcus Greyle': 1, 'Cobblestone': 1, 'Marston
Greyle!--has': 1, 'Fragonard': 1, 'S.S. _Araconda': 1, 'the Marston
Greyle': 1, 'any Marston Greyle': 1, 'Peter Chatfield!--they': 1,
'Bristol': 1, '_Margaret Sayers_.......': 1, 'ADELA CHATFIELD': 1, 'God':
1, 'Monty': 1, 'Simon Pure': 1, 'MARK GREY': 1, 'Marston Greyle!--never':
1, 'Guy Vickers': 1, 'Marconi': 1, 'Pike': 1, 'Mr. Vickers!--if': 1, "Mr.
Vickers'll": 1, 'Scott': 1, "Miss Greyle'll": 1, 'Mr. Vickers--': 1,
'Lor': 1, 'Great Scott': 1, 'Bassett Oliver!--the': 1, 'Zachary': 1,
'SCARVELL': 1, 'Addie Chatfield!': 1, 'ELKIN': 1, 'PETER CHATFIELD': 1,
'Mrs. Andrius': 1, 'Martin Andrius': 1, 'Elkin': 1})
```

Figure 7: *Person entities detected in J. S. Fletcher's Scarhaven Keep by the modified_deafult_pipeline once .*

Second, if an occurrence is detected only twice, it is still likely to be an error (see *Figure 8*).

```
'Mr. Richard Copplestone': 2, 'Waters': 2, 'Richard Copplestone': 2,
'this Marston Greyle': 2, 'Mr. Peter Chatfield': 2, 'Wooler': 2, 'Valentine
Greyle': 2, "Dan'l Ewbank": 2, 'Reverend Gilling': 2, 'Gad': 2,
'Mr. Coroner': 2, 'a Marston Greyle': 2, 'Montagu Gaines': 2, 'Stephen John
Greyle': 2, 'Altmores': 2, 'George': 2, 'Miss Audrey Greyle': 2,
'Scarhaven': 2, "Jim Spurge's": 2, 'High Nick': 2
```

Figure 8: *Person entities detected in J. S. Fletcher's Sarhaven Keep by the modified_deafult_pipeline twice .*

Third, if an occurrence is detected at least three times, we assume that it is a correctly detected entity (i.e., a true positive).

This post-processing measure might lead to discarding also true positives. However, discarding an infrequent occurrence does not necessarily lead to a great information loss – either the character is irrelevant, as it appears less than three times in the whole narrative, or the information is already encompassed in a more frequent occurrence, as demonstrated by the following example from *Scarhaven Keep*:

> The entity *Elkin* appears in the whole narrative only once; therefore, it is discarded. However, the entity *Mr. Elkin* appears in the narrative six times, and therefore it is kept. We leverage the information from the entities we keep in the following step to generate the possible aliases of a detected entity, e.g., we use *Mr. Elkin* to generate two aliases – *Mr. Elkin* and *Elkin*.

II. Generating aliases for entities occurring at least three times

In the previous step, we have retained occurrences occurring more than twice. With the assumption that these occurrences are true positives, we want them to be detected always, and in all their possible variations.

In order to do so, we create a list of occurrences to detect (*"occurrences_to_detect_list"*) that contains the entity that has been detected by *modified_default_pipeline* more than twice, and all its possible variations. E.g, the entity *Mrs. Valentine Greyle* has been detected four times, therefore it is appended to the list*,* together with its aliases *Valentine Greyle, Mrs. Valentine, Mrs. Greyle, Valentine* and *Greyle*.

We generate the possible aliases using *python nameparser*[14]. We parse the entities into their components, and then for each entity we append to the list:

- *title + first + last,*
- *first + last,*
- *title + first,*
- *title + last,*
- *first,*
- *last,*
- *title.*

The output of this step is an *occurrences_to_detect_list* of entities and their aliases. After some final tuning, this list will be used to generate patterns of occurrences to detect.

III. Final measures

We take three more steps to improve the performance. First, we remove from the *occurrences_to_detect_list* through exact matching all instances in genitives, as we want only the occurrences' lemmas to be included in the list. Second, we remove from the *occurrences_to_detect_list* all instances of determiners (see *Appendix D*)[15]. Sometimes spaCy NER incorrectly identifies determiners as a part of *person* entity, and thus *python nameparser* treats them as first name, as illustrated by example (7) from *Scarhaven Keep*.

[14] "Python Human Name Parser," Nameparser 1.1.2 Documentation, accessed April 4, 2023, https://nameparser.readthedocs.io/en/latest/index.html#.

[15] The list of determiners we remove is acquired from the following source: "Determiners (the, My, Some, This)," in *Cambridge Grammar*, accessed April 4, 2023, https://dictionary.cambridge.org/grammar/british-grammar/determiners-the-my-some-this.

(7) *this Marston Greyle (person)*

Third, we remove from the *occurrences_to_detect_list* the titles and honorifics followed by a full stop. It is not desirable to detect these when standing in isolation, as it complicates the unification process. Consider example (8) from *Scarhaven Keep:* the list does not contain the alias *Mr. Richard Copplestone*, but it contains the last name *Copplestone.* If the list included also honorifics followed by a full stop, the entity *Mr. Richard Copplestone* would be parsed by the pipeline into two entities (*Mr.* and *Copplestone*), each of which would in turn require unification.

(8) ***Mr.** (person) Richard **Copplestone** (person).*

The output of this step is the final *occurrences_to_detect_list* of entities and their aliases. In the following step, the list will be used to generate patterns to detect and tag as *person* entities.

4. Creation of patterns

We transform the items in the *occurrences_to_detect_list* into the actual *patterns* to be classified as *person* entities. The *patterns* include all possible aliases that are desirable to detect. The next step consists of loading the *patterns* into a spaCy NER blank English model (from now on called "*adapted_pipeline*"). We will use this *adapted_pipeline* to perform the NER task on the studied novels.

5. Evaluation of adapted_pipeline

We evaluate the performance of the *adapted_pipeline* on the same data as earlier, i.e., Chapter II of J. S. Fletcher's *Scarhaven Keep* (3,784 tokens). *Table 2* shows the performance of the *modified_default_pipeline* and the *adapted_pipeline*.

	modified_default_pipeline	**adapted_pipeline**
Precision	0.955	1
Recall	0.914	0.971
F1-score	0.934	0.986

Table 2: *Evaluation of the modified_default_pipeline and the adapted_pipeline.*

The *adapted_pipeline* performs well on named entities, their aliases and variations. By applying post-processing measures we have managed to further improve the performance in comparison to *modified_default_pipeline*. False negatives are caused by the fact that in case of infrequency the entities are not included in the *patterns.*

Detection of nominals and pronouns

The approach as described detects the occurrence of some of the nominals in the capitalized form, provided that they occurred at least three times in the position of title when run through the *modified_default_pipeline* and are not immediately followed by a full stop.

Our approach does not detect lower-case nominals (such as *porter* or *captain*) nor capitalized nominals, if they are not included in the *patterns*. Our approach also does not detect pronouns because we do not perform pronoun coreference resolution. According to Labatut and Bost (2019, 8) the detection and subsequent unification of nominals and pronouns is often omitted, as it is not considered much informative.

3.3.2 Identification of Character Occurrences' Interactions

As stated earlier, character interactions can be based on co-occurrences, conversations, mentions, direct actions or affiliations. We immediately discard the approach based on conversations because in novels a lot of interactions unfold in non-conversation parts. For the same reason, we decide against the approach based on mentions. We also decide against the direct-action-based approach because is complicated, as it requires the detection of actions, and identification of characters related to these actions. Finally, we discard the approach based on affiliations, as it is not a frequent one.

We choose to follow the approach based on co-occurrences. We are aware of its limitations, as described earlier. However, we do so for several reasons: firstly, according to Labatut and Bost (2019, 14) it is the most frequent approach, secondly, it is the easiest one to apply, thirdly, it contains the other approaches within itself.

The narrative unit we choose for this approach is one sentence, following the example of Lee and Yeung (2012), as such a choice does not result into a complete unit of text split into multiple parts. If a sentence contains at least two entities, the entities are combined into interaction pairs. Each entity is paired with all other entities detected in the same sentence. Below, we present in the example (9) a sentence from *Scarhaven Keep* that contains multiple entities, and the output of this stage for this sentence.

(9) But that very night, you, *Mr. Vickers*, and *Mr. Copplestone* and *Miss Greyle*, nearly stopped everything, and if *Andrius* and *Chatfield* hadn't carried you off, the scheme would have come to nothing. (chapter 31)

Detected Co-occurrences: (Mr. Vickers, Mr. Copplestone),
(Mr. Vickers, Miss Greyle),
(Mr. Vickers, Andrius),
(Mr. Vickers, Chatfield),
(Mr. Copplestone, Miss Greyle),
(Mr. Copplestone, Andrius),
(Mr. Copplestone, Chatfield),
(Miss Greyle, Andrius),
(Miss Greyle, Chatfield),
(Andrius, Chatfield)

The output of this stage is a list of interaction pairs (from now on called *"interaction_pairs"*) in chronological order.

3.3.3 Unification of Interacting Characters

The unification task in our case extends only to identifying the various aliases referring to the same character, i.e., we perform only *alias resolution*. We have adopted a rule-based approach, and we have drawn inspiration mostly from Ardunay and Sporleder's (2014) approach, which consists of three steps: 1. Human name parsing, 2. Gender assignation, and 3. Matching algorithm.

1. *Human Name Parsing*

Following the example of Ardunay and Sporleder (2014), first, we use *python nameparser*[16] to parse the interacting entities into their components (title, first name, last name).

2. *Gender Assignation*

Second, we define a gender generating function. Gender assignation is crucial for correct unification of characters., as illustrated by example (10) from *Scarhaven Keep*. If the entity appears only under the form of last name, in English it is impossible to assign the gender unequivocally. However, if the entity is preceded with a typically male or female title the gender can be assigned. By applying constraints (e.g., "do not unify entities

[16] "Python Human Name Parser."

that differ in gender"), we ensure that for example the entity *Mrs. Greyle* will not be unified with the male character called *Marston Greyle*, even though they share the same last name.

(10) *Greyle* *Mr. Marston Greyle*
Mrs. Valentine Greyle

We consider two criteria for gender assignation: 1. Presence of a typically male or a typically female honorific/title, 2. Presence of a typically male or typically female first name.

To assess the presence of a title indicative of gender, we have compiled two lists: one list containing typically male titles ("*male_titles_list"*) and one list containing typically female titles ("*female_titles_list"*).

The *male_titles_list* contains 22 items, such as *Lord, Mr., Sir* etc.; the *female_titles_list* contains 21 items, such as *Lady, Mrs., Miss* etc. (to see the respective lists, refer to *Appendix E*). Both lists were compiled manually from the Cambridge Dictionary: SMART vocabulary[17]. We added several extra items to both lists: *Uncle, Father, Brother* to the *male_titles_list*; *Aunt, Auntie, Mother, Sister* to the *female_titles_list*. We did so to account for "family titles" and for the titles and honorifics used in religious environments.

If the entity's title belongs to *male_titles_list*, or *female_titles_list*, it is assigned the *male* gender, or *female* gender respectively. If the title is ambiguous (e.g., *Doctor*) or not present, the second criteria is considered: the presence of a first name indicative of gender.

To assess the presence of a first name indicative of gender, we consider two lists acquired from SSA.gov web page[18]: one list containing the first thousand most popular male children names in the US in 2021 ("*male_names_list"*); one list containing the first thousand most popular female children names in the US in 2021 ("*female_names_list"*), to see the lists, refer to *Appendix F*.

[17] ""Royalty, Aristocracy & Titles - SMART Vocabulary Cloud with Related Words and Phrases," in *Cambridge Dictionary*, accessed April 5, 2023, https://dictionary.cambridge.org/topics/society/royalty-aristocracy-and-titles/.

[18] We acquired both list from "Popular Baby Names," Social Security Administration(.gov), accessed April 5, 2023, https://www.ssa.gov/cgi-bin/popularnames.cgi.

If the entity's first name belongs to both lists, its gender remains *unknown*. If it belongs only to the *male_names_list*, its gender is tagged as *male*; if it belongs only to the *female_names_list*, its gender is tagged as *female*. If the entity cannot be assigned a gender based on the two criteria (i.e., based on title or first name), its gender is tagged as *unknown*.

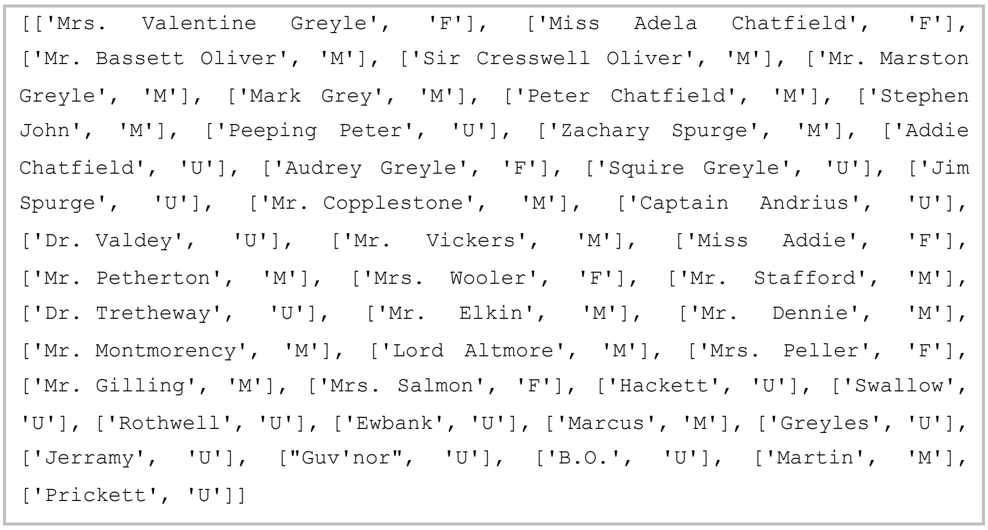
3. Matching algorithm

Third, we define the matching algorithm. In order to match the interacting occurrences with the character they refer to, first, we need to generate a list containing all possible referents. Then, we generate a set of rules to unify the various occurrences with their referents.

I. Creating a list of referents

During this step, we create a list of referents ("*referents_list"*) the various aliases and variations refer to. In order to unify the characters, we will match their aliases against the referents in the *referents_list*.

The output of the Stage 1 (character occurrences detection) is a chronological sequence of detected occurrences. We create a list of aliases ("*aliases_list*"), where each occurrence detected in Stage 1 is included once. Then for each alias in *aliases_list*, from the least ambiguous to the most ambiguous, we check whether its referent is included in the *referents_list*. If not, we append the alias with the indication of its gender to the *referents_list. Figure 9* illustrates the final *referents_list* for *Scarhaven Keep.*



```
[['Mrs. Valentine Greyle', 'F'], ['Miss Adela Chatfield', 'F'],
['Mr. Bassett Oliver', 'M'], ['Sir Cresswell Oliver', 'M'], ['Mr. Marston
Greyle', 'M'], ['Mark Grey', 'M'], ['Peter Chatfield', 'M'], ['Stephen
John', 'M'], ['Peeping Peter', 'U'], ['Zachary Spurge', 'M'], ['Addie
Chatfield', 'U'], ['Audrey Greyle', 'F'], ['Squire Greyle', 'U'], ['Jim
Spurge', 'U'], ['Mr. Copplestone', 'M'], ['Captain Andrius', 'U'],
['Dr. Valdey', 'U'], ['Mr. Vickers', 'M'], ['Miss Addie', 'F'],
['Mr. Petherton', 'M'], ['Mrs. Wooler', 'F'], ['Mr. Stafford', 'M'],
['Dr. Tretheway', 'U'], ['Mr. Elkin', 'M'], ['Mr. Dennie', 'M'],
['Mr. Montmorency', 'M'], ['Lord Altmore', 'M'], ['Mrs. Peller', 'F'],
['Mr. Gilling', 'M'], ['Mrs. Salmon', 'F'], ['Hackett', 'U'], ['Swallow',
'U'], ['Rothwell', 'U'], ['Ewbank', 'U'], ['Marcus', 'M'], ['Greyles', 'U'],
['Jerramy', 'U'], ["Guv'nor", 'U'], ['B.O.', 'U'], ['Martin', 'M'],
['Prickett', 'U']]
```

Figure 9: *Referents_list for Scarhaven Keep*

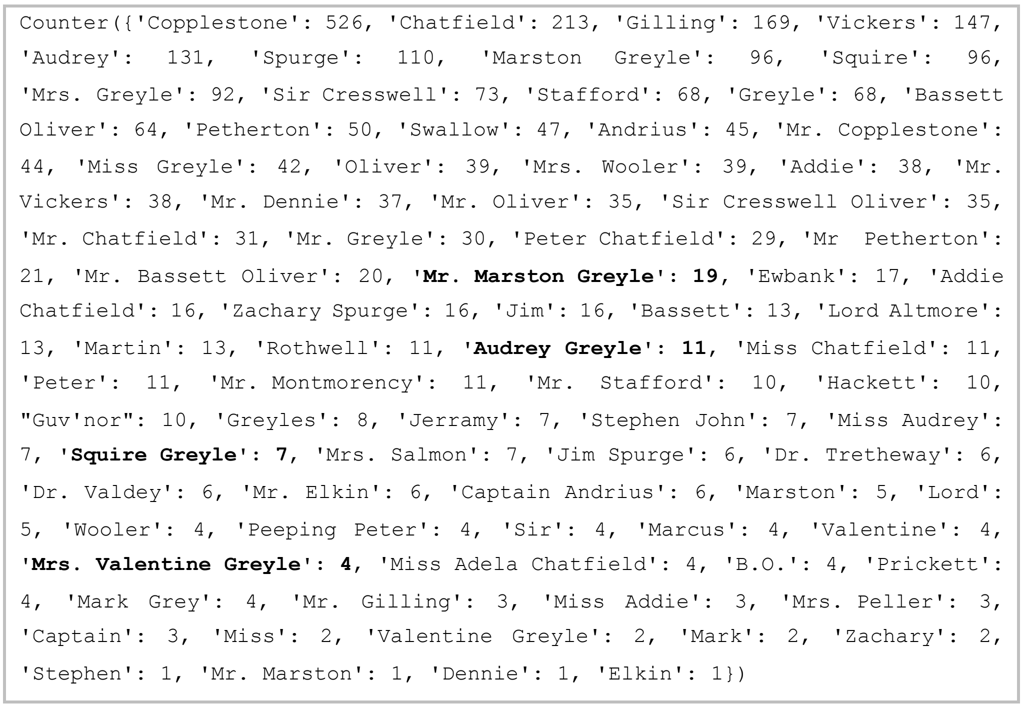
.

II. *Rule-based matching: unification of interacting occurrences with their referents*

The last phase of the unification process consists of unifying the interacting occurrences (output of Stage 2, items in *interaction_pairs*) with their referents from *referents_list* based on rule-based matching. We have generated a set of rules to account for possible scenarios, from least to most ambiguous, taking into account string similarity and gender compatibility.

The unification of some interacting occurrences is unequivocal (e.g., *Marston Greyle* will be unequivocally unified with *Mr. Marston Greyle*, based on the same full name). In other cases, the unification is not so straight-forward. We consider especially problematic for unification the entities that appear isolated in the form of *last name*, like the commonly used last name *Greyle* in *Scarhaven Keep* that can refer to four referents (11).

(11) *Greyle* *Mr. Marston Greyle*
Mrs. Valentine Greyle
Audrey Greyle
Squire Greyle



```
Counter({'Copplestone': 526, 'Chatfield': 213, 'Gilling': 169, 'Vickers': 147,
'Audrey': 131, 'Spurge': 110, 'Marston Greyle': 96, 'Squire': 96,
'Mrs. Greyle': 92, 'Sir Cresswell': 73, 'Stafford': 68, 'Greyle': 68, 'Bassett
Oliver': 64, 'Petherton': 50, 'Swallow': 47, 'Andrius': 45, 'Mr. Copplestone':
44, 'Miss Greyle': 42, 'Oliver': 39, 'Mrs. Wooler': 39, 'Addie': 38, 'Mr.
Vickers': 38, 'Mr. Dennie': 37, 'Mr. Oliver': 35, 'Sir Cresswell Oliver': 35,
'Mr. Chatfield': 31, 'Mr. Greyle': 30, 'Peter Chatfield': 29, 'Mr  Petherton':
21, 'Mr. Bassett Oliver': 20, 'Mr. Marston Greyle': 19, 'Ewbank': 17, 'Addie
Chatfield': 16, 'Zachary Spurge': 16, 'Jim': 16, 'Bassett': 13, 'Lord Altmore':
13, 'Martin': 13, 'Rothwell': 11, 'Audrey Greyle': 11, 'Miss Chatfield': 11,
'Peter': 11, 'Mr. Montmorency': 11, 'Mr. Stafford': 10, 'Hackett': 10,
"Guv'nor": 10, 'Greyles': 8, 'Jerramy': 7, 'Stephen John': 7, 'Miss Audrey':
7, 'Squire Greyle': 7, 'Mrs. Salmon': 7, 'Jim Spurge': 6, 'Dr. Tretheway': 6,
'Dr. Valdey': 6, 'Mr. Elkin': 6, 'Captain Andrius': 6, 'Marston': 5, 'Lord':
5, 'Wooler': 4, 'Peeping Peter': 4, 'Sir': 4, 'Marcus': 4, 'Valentine': 4,
'Mrs. Valentine Greyle': 4, 'Miss Adela Chatfield': 4, 'B.O.': 4, 'Prickett':
4, 'Mark Grey': 4, 'Mr. Gilling': 3, 'Miss Addie': 3, 'Mrs. Peller': 3,
'Captain': 3, 'Miss': 2, 'Valentine Greyle': 2, 'Mark': 2, 'Zachary': 2,
'Stephen': 1, 'Mr. Marston': 1, 'Dennie': 1, 'Elkin': 1})
```

Figure 10: *Frequency of the occurrences in Scarhaven Keep*

Our approach is based on rules and exact matching, it does not take into account the context of the occurrence. If an entity can refer to multiple referents, we make a simplification, and we assume that it refers to the most frequently occurring of these referents. *Figure 10* (above) shows how many times the occurrences in our illustrative novel, *Scarahven Keep*, have been detected.

Therefore, for example (11), the occurrence *Greyle* would be unified with the most frequent referent, i.e., *Mr. Marston Greyle*.

This is a simplification which is highly error-prone, as we consider only the frequency of the referents. The referents are the least ambiguous variations of the character names; not necessarily the most frequent ones (e.g., *Mr. Marston Greyle* is detected in the narrative 19 times, whereas its alias *Marston Greyle* is detected 96 times). However, in order to simplify the process, we settle for this solution to the problem of multiple referents – it is of outmost importance to unify the ambiguous entity with exactly one referent from the *referents_list*.

Further Notes on the Unification Process

First, we want to state clearly that the unification process is not error-free. We have designed a set of rules based on string similarity and gender compatibility. The scenarios not accounted for in the rules are not dealt with. As we have generated the rules ourselves, it is possible that we have unconsciously omitted rules for some possible variations of aliases, or that some possible referents might be missing from the *referents_list*.

Second, we do not attempt to resolve the unification of nicknames and hypocorisms. To do so, we would need an immense external source with their list. Given the already complexity of the code, we decide against further modification of the unification process. This decision leads to detecting multiple referents for a unique character, as is the case in the following example (12).

(12)	**Referent**	**Character**
	Miss Adela Chatfield	Adela Chatfield
	Addie Chatfield	Adela Chatfield
	Mr. Basset Oliver	Bassett Oliver
	B.O.	Bassett Oliver

Third, we exclude self-loops from the final edge-list (list of unified interacting character pairs), i.e., if an entity is detected in the same sentence more than once, the pair consisting of (*entity, entity*) is excluded from consideration.

The output of the Stage 3 is a chronological sequence of unified characters interacting within the *interaction_pairs*. *Table 3* shows the output of the unification stage for the first five interaction pairs of *Scarhaven Keep.*

	Before unification	**After unification**
1	(Mr. Oliver, Stafford)	(Mr. Bassett Oliver, 'Mr. Stafford)
2	(Stafford, Mr. Oliver)	(Mr. Stafford, Mr. Bassett Oliver)
3	(Mr. Oliver, Mr. Copplestone)	(Mr. Bassett Oliver, Mr. Copplestone)
4	(Stafford, Copplestone)	(Mr. Stafford, Mr. Copplestone)
5	(Mr. Copplestone, Rothwell)	(Mr. Copplestone, 'Rothwell)

Table 3: *First five interacting pairs of Fletcher's Scarhaven Keep before and after unification.*

3.3.4 Graph Extraction

In this section we describe the last stage of the network extraction process, the extraction of the network proper. The nodes of the resulting graphs are the individual characters (referents); a link is created between any two characters co-occurring in the same sentence. The extracted graphs are undirected, unweighted, unsigned and static.

The edges are undirected, as the co-occurrence-based approach to interaction detection intrinsically leads to the extraction of bilateral interactions. They are also unweighted, as we do not measure a score of any type during the extraction process. We could for example keep track of the frequency of the detected interactions, however, at this stage we decide to keep the process simple. Due to the error-proneness of the unification stage, we could incidentally assign higher or lower score to any given interacting pair of characters. It follows from the above-mentioned that we do not address the polarity of the relationships either.

We extract static networks - we extract one graph for the whole narrative of each analysed novel. It is out of the scope of this thesis to extract dynamic networks of the studied novels. However, we believe that the dynamic aspect of the character network building process could be of interest to the authorship attribution task, and we recommend its exploration for future works.

In order to extract the graphs proper, we use the output of Stage 3, the edge-list of interacting characters after unification. To create the graphs, we use *NetworkX*[19], a Python package for manipulation and exploration of complex networks. For graph visualization, we use the free open-source software Gephi[20], which is a standard tool used in the field of graph visualization.

The output of this stage are 75 distinct character networks extracted from the analysed novels written by the five selected authors. To illustrate the output of Stage 4, the following *Figures* (*11–16*) show the graphs extracted from the illustrative novel, *Scarhaven Keep*, and from *text_1* of the respective authors (see *Appendix A*).

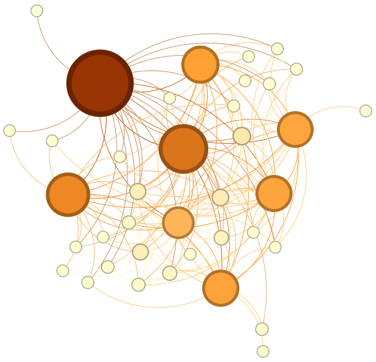


Figure 11: *J. S. Fletcher, Scarhaven Keep*

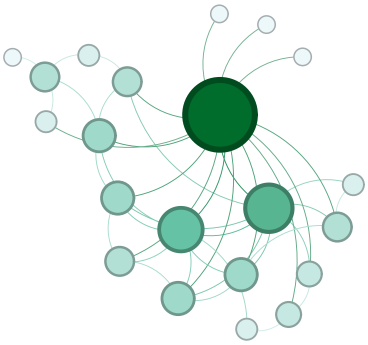


Figure 12: *Author_1, text_1*

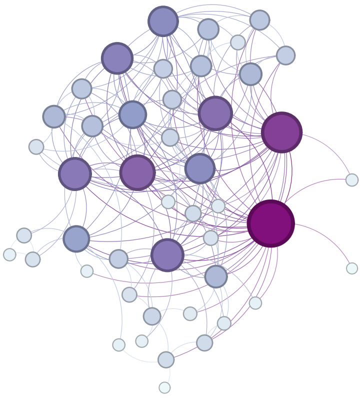


Figure 13*: Author_2, text_1*

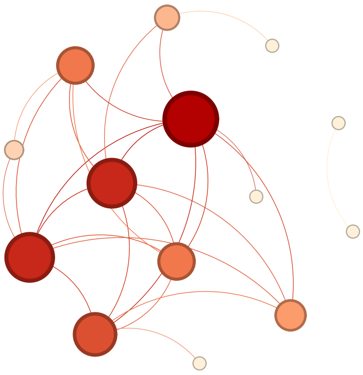


Figure 14: *Author_3, text_1*

[19] "NetworkX," NetworkX Documentation, accessed April 10, 2023, https://networkx.org/.
[20] "Gephi - The Open Graph Viz Platform," Gephi, accessed April 10, 2023, https://gephi.org/.

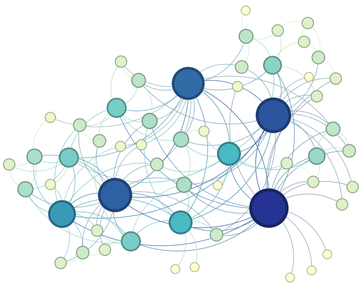


***Figure 15:** Author_4, text_1*

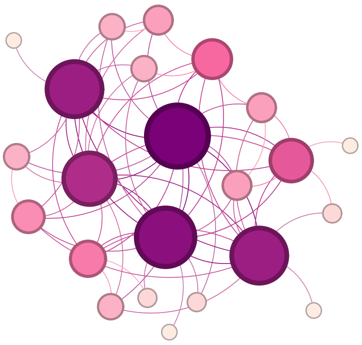


***Figure 16:** Author_5, text_1*

3.4 Extraction of Quantitative Character Network Features

As described in the *Theoretical Part*, there are many quantitative characteristics that can be calculated for any graph. We use *NetworkX*[21] to calculate the following eight different network features for each of the 75 extracted networks: 1. Number of Nodes, 2. Number of Edges, 3. Modularity, 4. Average Clustering Coefficient, 5. Average Degree Centrality, 6. Average Betweenness Centrality, 7. Average Closeness Centrality, 8. Average Eigenvector Centrality.

Average centrality measures (e.g., average betweenness centrality) are not often used as global graph indices, because of the possible presence of outliers in the data. Outliers may divert the value of an average variable in one direction, misleading the interpretation of the results. In such cases, it is preferable to use median of the given variable. However, using median in our case would be problematic (for example, in many cases of betweenness centrality the median of the network would be equal to zero, e.g., for *Author_1,* it is the case for five networks). Therefore, we settle for the average value of the centrality measures, as in our case the possible presence of outliers does not mislead the interpretation of the results - of the outmost importance is to extract a quantitative feature from each graph through the same calculation[22].

[21] "NetworkX."

[22] All network features are extracted using *NetworkX* package, and the way it is computed is described in the *NetworkX* documentation. Of special importance is to consider that *NetworkX* returns the normalized values of the centrality measures.

The documentation for the respective network features is available at:

1. ""Degree_centrality," NetworkX 3.1 Documentation, accessed April 10, 2023, https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.degree_centrality.html.

Initially, we choose to extract number of nodes and edges, as these are the basic parameters of each graph. Then we extract modularity and average clustering coefficient, because we believe that the examination of these two characteristics could be interesting for authorship attribution. Finally, we extract the four centrality measures, as we believe that it will be interesting to examine the way that different authors distribute power within their networks.

Tables 4-8 show the values of the eight features extracted from the character networks built by *Author_1*, *Author_2*, *Author_3*, *Auhtor_4*, and *Author_5* respectively.

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	21	43	0.271	0.355	0.205	0.059	0.488	0.18
text_2	20	53	0.155	0.497	0.279	0.048	0.558	0.191
text_3	28	58	0.368	0.625	0.153	0.053	0.439	0.15
text_4	39	55	0.412	0.413	0.074	0.034	0.457	0.126
text_5	28	60	0.325	0.483	0.159	0.041	0.498	0.16
text_6	29	66	0.311	0.531	0.163	0.04	0.495	0.154
text_7	31	69	0.317	0.62	0.148	0.038	0.489	0.151
text_8	37	95	0.301	0.497	0.143	0.033	0.478	0.134
text_9	30	51	0.353	0.538	0.117	0.042	0.471	0.148
text_10	36	94	0.29	0.512	0.149	0.032	0.488	0.135
text_11	31	75	0.316	0.572	0.161	0.034	0.517	0.152
text_12	31	79	0.322	0.567	0.17	0.034	0.516	0.153
text_13	33	68	0.4	0.58	0.129	0.035	0.491	0.141
text_14	31	82	0.31	0.624	0.176	0.037	0.495	0.146
text_15	43	105	0.357	0.562	0.116	0.032	0.446	0.12

Table 4: *Network features extracted from networks built by Author_1.*

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	46	191	0.269	0.598	0.185	0.025	0.489	0.118
text_2	48	247	0.276	0.66	0.219	0.021	0.524	0.123
text_3	69	324	0.316	0.651	0.138	0.016	0.499	0.097
text_4	45	132	0.274	0.547	0.133	0.026	0.489	0.119
text_5	78	447	0.288	0.582	0.149	0.015	0.486	0.088
text_6	67	334	0.263	0.604	0.151	0.017	0.492	0.098
text_7	57	222	0.306	0.539	0.139	0.02	0.485	0.105
text_8	24	87	0.212	0.726	0.315	0.036	0.578	0.179
text_9	71	285	0.371	0.591	0.115	0.015	0.458	0.092
text_10	14	29	0.175	0.673	0.319	0.062	0.592	0.241
text_11	54	267	0.231	0.633	0.187	0.019	0.519	0.11
text_12	76	246	0.408	0.557	0.086	0.017	0.459	0.09
text_13	13	34	0.122	0.757	0.436	0.054	0.651	0.252
text_14	51	197	0.253	0.533	0.155	0.022	0.492	0.113
text_15	83	344	0.391	0.526	0.101	0.017	0.435	0.085

Table 5: *Network features extracted from networks built by Author_2.*

[2] "Betweenness_centrality," NetworkX 3.1 Documentation, accessed April 10, 2023, https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.betweenness_centrality.html.

[3] "Closeness_centrality," NetworkX 3.1 Documentation, accessed April 10, 2023, https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.closeness_centrality.html.

[4] "Eigenvector_centrality," NetworkX 3.1 Documentation, accessed April 10, 2023, https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.eigenvector_centrality.html.

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	14	26	0.183	0.431	0.286	0.052	0.422	0.215
text_2	25	64	0.275	0.565	0.213	0.046	0.502	0.166
text_3	22	58	0.2	0.564	0.251	0.046	0.537	0.181
text_4	15	37	0.217	0.616	0.352	0.062	0.573	0.226
text_5	24	64	0.272	0.793	0.232	0.037	0.562	0.178
text_6	10	20	0.071	0.406	0.444	0.083	0.624	0.288
text_7	26	49	0.317	0.486	0.151	0.047	0.486	0.163
text_8	30	110	0.286	0.718	0.253	0.038	0.503	0.149
text_9	15	53	0.198	0.818	0.505	0.038	0.685	0.243
text_10	20	49	0.216	0.736	0.258	0.045	0.566	0.201
text_11	14	39	0.151	0.663	0.429	0.052	0.635	0.245
text_12	15	28	0.238	0.381	0.267	0.091	0.483	0.217
text_13	33	128	0.279	0.661	0.242	0.036	0.493	0.146
text_14	24	107	0.138	0.759	0.388	0.03	0.62	0.181
text_15	13	53	0.048	0.821	0.679	0.031	0.772	0.262

Table 6: *Network features extracted from networks built by Author_3.*

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	56	152	0.386	0.485	0.099	0.027	0.419	0.103
text_2	47	168	0.324	0.517	0.155	0.03	0.439	0.111
text_3	58	285	0.245	0.574	0.172	0.017	0.476	0.107
text_4	50	130	0.362	0.423	0.106	0.028	0.441	0.11
text_5	47	160	0.32	0.553	0.148	0.028	0.455	0.119
text_6	40	112	0.333	0.504	0.144	0.033	0.46	0.126
text_7	42	166	0.248	0.542	0.193	0.025	0.518	0.128
text_8	82	369	0.29	0.584	0.111	0.015	0.463	0.084
text_9	64	314	0.245	0.533	0.156	0.018	0.486	0.101
text_10	32	63	0.374	0.408	0.127	0.048	0.427	0.14
text_11	41	99	0.407	0.446	0.121	0.035	0.433	0.126
text_12	55	272	0.246	0.599	0.183	0.021	0.485	0.109
text_13	47	93	0.417	0.38	0.086	0.034	0.409	0.114
text_14	47	100	0.41	0.458	0.093	0.033	0.413	0.113
text_15	38	135	0.246	0.687	0.192	0.026	0.533	0.131

Table 7: *Network features extracted from networks built by Author_4.*

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	23	69	0.184	0.62	0.273	0.043	0.548	0.178
text_2	30	73	0.276	0.425	0.168	0.039	0.493	0.147
text_3	33	107	0.302	0.553	0.203	0.035	0.497	0.146
text_4	26	105	0.182	0.744	0.323	0.029	0.6	0.175
text_5	44	143	0.296	0.477	0.151	0.029	0.466	0.121
text_6	61	187	0.37	0.537	0.102	0.02	0.382	0.094
text_7	41	102	0.302	0.581	0.124	0.029	0.478	0.124
text_8	20	41	0.247	0.605	0.216	0.051	0.536	0.195
text_9	55	166	0.339	0.48	0.112	0.025	0.398	0.1
text_10	34	127	0.217	0.579	0.226	0.03	0.522	0.142
text_11	16	37	0.223	0.718	0.308	0.054	0.588	0.218
text_12	19	55	0.182	0.605	0.322	0.042	0.6	0.209
text_13	50	176	0.301	0.703	0.144	0.022	0.499	0.113
text_14	20	62	0.221	0.714	0.326	0.041	0.59	0.2
text_15	48	128	0.309	0.461	0.113	0.026	0.415	0.107

Table 8: *Network features extracted from networks built by Author_5.*

3.5 Are Network Features and the Length of the Novel Correlated?

As mentioned earlier, it would have been desirable to select 75 novels of the same length. However, this condition proved to be difficult to meet, as authors frequently vary in the length of the novels – not only the authors differ one from the other in terms of the length of the novels, but also the novels written by the one and the same author differ in their lengths considerably (see *Appendix A*).

We make a presupposition that the way a character network is built will vary according to the length of the novel. To prove or discard this presupposition, we calculate

Spearman's rank correlation coefficient for the individual network features and the lengths of the novels.

Spearman's Rank Correlation Coefficient

Spearman's rank correlation coefficient, denoted by ρ or r_s, measures (nonparametrically) rank correlations between the variable X and the variable Y. It measures the strength and direction of the relationship between the two variables. The coefficient assesses monotonic relationships between the two variables: if $\rho = 1$ or $\rho = -1$, there is a perfect monotone relationship between the two variables, if $\rho = 0$, there is no correlation between the two variables. More on Spearman's ρ for example in Akoglu (2018) or Prion and Haerling (2014).

We used the online tool available at Statskingdom.com[23] to calculate Spearman's ρ for each pair of variables (i.e., for length and the respective network feature). The values of Spearman's ρ, p-values, and covariances can be seen in *Table 9*.

	Spearman's ρ	**p-value**	**Covariance**	**Statistic**
Number of Nodes (N)	0.663	0.000	314.642	7.56
Number of Edges (L)	0.674	0.000	320.264	7.802
Modularity	0.238	0.040	112.851	2.09
Average Clustering Coefficient	-0.125	0.287	-59.203	-1.073
Average Degree Centrality	-0.386	0.001	-183.581	-3.58
Average Betweenness Centrality	-0.640	0.000	-303.784	-7.108
Average Closeness Centrality	-0.366	0.001	-174.054	-3.365
Average Eigenvector Centrality	-0.647	0.000	-307.338	-7.25

Table 9: *Results of the calculation of Spearman's rank correlation coefficient for the variable length of the novels and the respective network features.*

The results of Spearman's ρ correlation indicate that there is a significant large positive relationship between the length of a novel and the number of nodes and edges; there is a significant small positive relationship between the length and the modularity; there is a significant very small negative relationship between the length of the novel and all given centrality measures. The only feature that does not correlate with the length of the novel is the average clustering coefficient; the relationship between the length and the average clustering coefficient is very small and statistically not significant.

We can conclude that our presupposition was correct, and that network features and the length of the novel indeed correlate. In order to address this issue, we shorten all

[23] "Correlation Coefficient Calculator - Including the Covariance and Calculation Steps," Statskingdom.com, accessed April 25, 2023, https://www.statskingdom.com/correlation-calculator.html.

sample novels to the same length of first 50,000 tokens. We set this limit by considering the length of the shortest novel (*Author_5, text_10*, length (tokens) = 50,997). This methodological choice is not ideal. By shortening the novels using a physical criterion, we ignore the author's plan of character distribution over the plot evolution.

The issue results into the following: the shortest analysed novel *(Author_5*, *text_10*, 50,997 tokens) is likely to have extracted an almost complete character network, whereas the longest novel (*Author_2*, *text_2*, 128,571 tokens) will be shortened to approximately 40% of its length. Character interactions introduced in the rest of the novel will be lost.

An alternative solution to the presented problem would be chunking the novels into smaller chunks and then "reconstructing" the novels with randomly selected chunks up to the final length of 50,000 tokens. Nevertheless, we settle ourselves for extracting character networks from the first 50,000 tokens of each novel; as we do not have any evidence that shortening the novels will indeed influence the results. To see the values of the eight selected network features extracted from the shorted versions of the novels, go to in *Appendix G*.

3.6 Binary Logistic Regression: An Introduction

The selected statistical method we use to estimate the probability of the authorship of a given author is binary logistic regression.

Binary logistic regression is a statistical method used for classification and probability prediction. It estimates the probability of an event occurring/not occurring (dependent variable, criterion) based on the given dataset of observed variables (independent variables, predictors). The dependent variable in the case of binary logistic regression is dichotomous, i.e., it takes only two possible values (e.g., 1 or 0; event occurring or event not occurring).

Logistic regression is used to estimate the probability of an event occurring, i.e., $P(Y = 1|x)$, based on the values of the independent variables x. The probability $P(Y = 1|x)$ is a value between 0 and 1, and it can be expressed by formula (1), where x are the values of independent variables, β_0 is the intercept and β_i are the regression coefficients.

$$P(Y=1|x) = P^{(x)} = \frac{1}{1+e^{-(\beta_0+\sum_{i=1}\beta_i x_i)}} \qquad (1)$$

If the regression coefficient β_i of an independent variable is equal to zero, we can conclude that the variable does not influence the probability of the event $P(Y = 1|x)$. If β_i is positive, the variable influences the probability of $P(Y = 1|x)$ positively, if β_i is negative, the variable influences the probability of $P(Y = 1|x)$ negatively. More on logistic regression in LaValley (2008), DeMaris (1995), Sperandei (2014) or Stoltzfus (2011).

In our case, we shall use logistic regression models to determine for each pair of analysed authors whether the network features of their character networks contribute to the authorship attribution task. The dependent variable is simply whether a given author is the author of the novel of concern or not. The selection of the independent variables is described in detail in the following section.

3.6.1 The Selection of Independent Variables

At this stage of the analysis, we have extracted eight network features (see above *Table 4-8*). However, our sample size $n = 75$ requires us to discard one of these features, as performing any further data exploration is conditioned by having at least 10 observations per variable. DeCoster (1998, 4) speaks about this limit for performing Exploratory Factor Analysis, which we do in the following step.

We assume that the number of nodes and the number of edges of a graph correlate; and we calculate Spearman's ρ for these two variables[24]. As demonstrated by the results in *Table 10*, we conclude that there is indeed a statistically significant large positive relationship between them. Therefore, we remove the variable number of edges from all following considerations, and we work with the seven remaining network features.

	Spearman's ρ	**p-value**	**Covariance**	**Statistic**
Value	0.889	0.000	421.764	16.577

Table 10: *Spearman correlation for the variables number of nodes and number of edges.*

However, we need to restrict the number of the features further. To estimate the probability of authorship for each pair of authors, we use logistic regression. The "rule of thumb" of logistic regression is a one in ten rule, i.e., for each variable we analyse there

[24] For the calculation we use the online tool available at "Correlation Coefficient Calculator - Including the Covariance and Calculation Steps," Statskingdom.com, accessed April 29, 2023, https://www.statskingdom.com/correlation-calculator.html.

should be at least 10 events per outcome; the least common outcome determines the number of variables (Stoltzfus 2011, 1101). In our case, there are 15 outcomes for each category for each model (15 novels by *Author_X,* 15 novels by *Author_Y*).

Following the one in ten rule, our sample size would allow us to study the influence of 1.5 features. However, according to Vittinghoff and McCulloch (2007, 717), it is possible to relax the above-described rule to 5 events. We do so, and this allows us to study the influence of three independent variables on the event of interest.

We need to bear in mind that by relaxing the number of events per variable, the probability that a significant result is influenced by coincidence, irrelevant combination of features, and by specific circumstances, increases. Consequently, it is harder to assess the relevance of a given feature, and the resulting model is not suited for the analysis of unknown novels and for making important decisions.

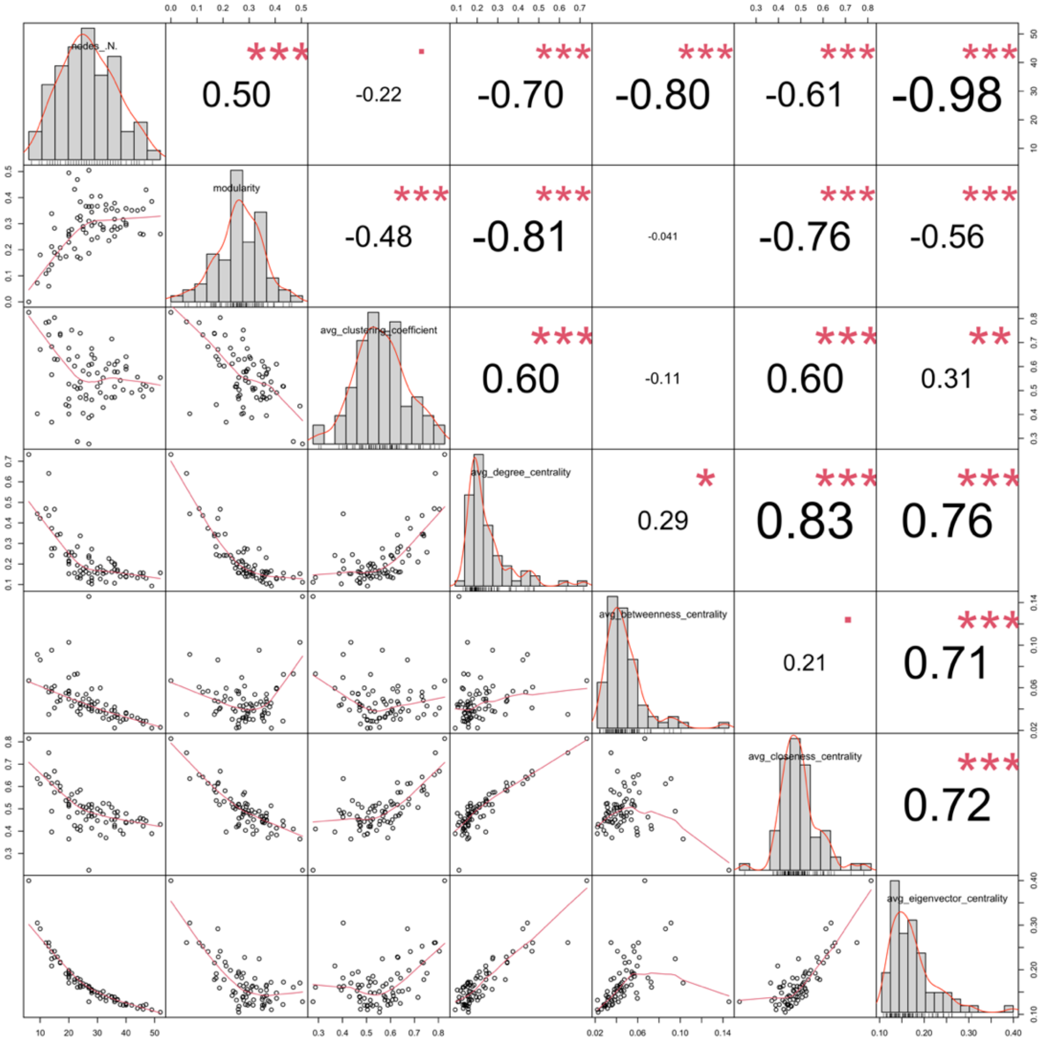


Figure 17: *Correlation matrix of the network features.*

In order to select from our set of seven network features the three features of interest, we need to assess the correlations among them. To do so, we extract their correlation matrix[25] (see above *Figure 17*); and we perform Exploratory Factor Analysis.

3.6.2 Factor Analysis

In our case, we perform Exploratory Factor Analysis (EFA). EFA is a statistical method used to identify underlying constructs (latent variables, factors) that influence the observed variables. Using EFA, the observed variables can be grouped according to shared variance into different clusters represented by the factors.

Factor analysis builds upon the Common Factor Model (see DeCoster 1998, 1). The idea of this model is that all observed variables are partially influenced by underlying factors. The link between a factor and the observed variables varies from variable to variable in the strength of the influence. The aim is to identify an underlying factor behind the variables that are highly correlated and separate them from the variables that are influenced by a different factor. More on Factor Analysis in DeCoster (1998) or Yong and Pearce (2013).

To perform EFA, we have used the online tool *QUITA*[26] We have grouped the seven observed variables into three groups of underlying latent variables. *Table 11* presents the values of factor loadings which indicate the strength of the influence between observed variables and the factors, i.e., how highly the seven observed variables correlate with the factors. *Figure 18* shows the results of the factor analysis.

	Factor 1	**Factor 2**	**Factor 3**
Number of Nodes (N)	-0.59	**-0.68**	-0.11
Modularity	**-0.87**	0.05	-0.24
Average Clustering Coefficient	0.41	-0.11	**0.90**
Average Degree Centrality	**0.83**	0.28	0.38
Average Betweenness Centrality	-0.04	**0.95**	-0.14
Average Closeness Centrality	**0.88**	0.06	0.38
Average Eigenvector Centrality	**0.79**	0.55	0.19

Table 11: *Factor loadings.*

[25] We extract the correlation matrix in *R* ("R: The R Project for Statistical Computing," R Project, accessed April 26, 2023, https://www.r-project.org/); using the code given in *Appendix H*.

[26] "QUITA Online," Quita, accessed April 26, 2023, https://kol.ff.upol.cz/quita/.

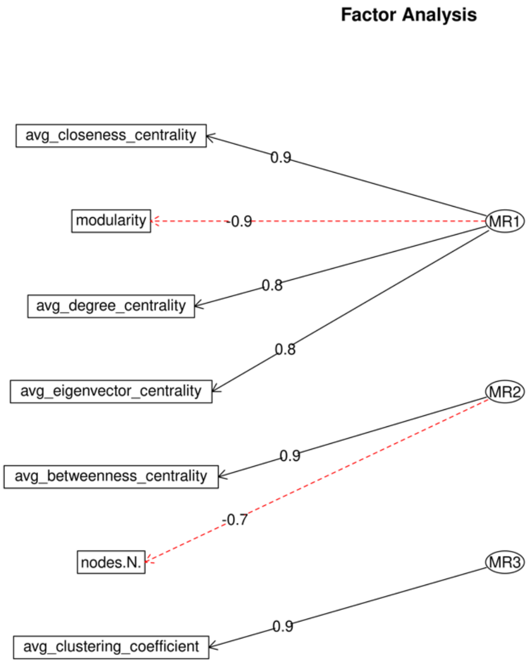


Figure 18: *Results of Exploratory Factor Analysis*

Factor 3 is the underlying construct that influences only one observed variable: average clustering coefficient. Its selection as the first independent variable is therefore straightforward.

Factor 2 is the underlying construct that influences two observed variables: average betweenness centrality and number of nodes. We select for further analysis the variable average betweenness centrality, as the strength of the influence between the given variable and the given factor is higher (see *Table 11*).

Factor 1 is the underlying construct that influences four observed variables. The strength of the influence is highest in the case of modularity and average closeness centrality (see *Table 11*). We can select only one of the two variables, and we have decided to consider the influence of average closeness centrality on the event of concern, as we are interested in the distribution of power within the networks.

Using EFA, we have selected three final independent relatively uncorrelated variables: average clustering coefficient, average betweenness centrality and average closeness centrality. In the next step, we will explore the influence of these three independent variables on the dependent variable ("*Author_X* is the author") by performing binary logistic regression analysis.

3.7 Binary Logistic Regression Applied to Our Data

In our case, we shall use logistic regression models to determine for each pair of analysed authors whether the quantitative network features of their fiction character networks contribute to the authorship attribution task or not.

The following *Table 12-16* show summary of the data's main descriptive characteristics (to see all values of the three selected features for the respective authors, go to *Appendix I*).

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
Min.	0.386	0.032	0.413
1st Qu.	0.451	0.043	0.455
Median	0.510	0.044	0.494
Mean	0.496	0.047	0.482
3rd Qu.	0.545	0.051	0.511
Max.	0.610	0.07	0.537

Table 12*: Summary of main descriptive characteristics: minimum, first quantile, median, mean, third quantile and maximum value of the input data by* ***Author_1****.*

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
Min.	0.472	0.022	0.388
1st Qu.	0.510	0.028	0.425
Median	0.589	0.032	0.441
Mean	0.605	0.037	0.49
3rd Qu.	0.678	0.040	0.492
Max.	0.826	0.067	0.815

Table 13*: Summary of main descriptive characteristics: minimum, first quantile, median, mean, third quantile and maximum value of the input data by* ***Author_2****.*

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
Min.	0.368	0.035	0.397
1st Qu.	0.537	0.042	0.487
Median	0.675	0.049	0.565
Mean	0.627	0.055	0.556
3rd Qu.	0.738	0.058	0.613
Max.	0.804	0.095	0.751

Table 14*: Summary of main descriptive characteristics: minimum, first quantile, median, mean, third quantile and maximum value of the input data by* ***Author_3.***

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
Min.	0.277	0.029	0.225
1st Qu.	0.481	0.034	0.396
Median	0.518	0.04	0.452
Mean	0.507	0.055	0.434
3rd Qu.	0.584	0.066	0.488
Max.	0.650	0.146	0.518

Table 15*: Summary of main descriptive characteristics: minimum, first quantile, median, mean, third quantile and maximum value of the input data by* ***Author_4****.*

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
Min.	0.405	0.028	0.404
1st Qu.	0.478	0.034	0.469
Median	0.602	0.045	0.487
Mean	0.565	0.043	0.507
3rd Qu.	0.630	0.051	0.553
Max.	0.731	0.062	0.617

Table 16: *Summary of main descriptive characteristics: minimum, first quantile, median, mean, third quantile and maximum value of the input data by* ***Author_5.***

The independent variables are the standardized values[27] of three different network features of a given character network, namely:

$\boldsymbol{x_1}$ … average clustering coefficient,
$\boldsymbol{x_2}$ … average betweenness centrality,
$\boldsymbol{x_3}$ … average closeness centrality,

if the independent variables are values extracted from a graph built by *Author_X* (i.e., "*Author_X* is the author"), the dependent variable is assigned value "1"; if the independent variables are values extracted from a graph built by *Author_Y* (i.e., "*Author_*X is not the author"), the dependent variable is assigned value "0".

We compare two models:

- the baseline model without any independent variables (with $ln(odds) = \beta_0$),
- the model with independent variables (with $ln(odds) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$),

where β_0 is the intercept (a constant), and β_1, β_2, β_3 are the regression coefficients calculated for the respective independent variables.

H_0 (null hypothesis) is defined as follows:

There is no statistically significant difference between the model without the independent variables and the model with the independent variables in the prediction of the event "*Author_X* is the author". In other words, there is no significant difference in the influence of the analysed quantitative graph properties for *Author_X* and for *Author_Y*.

H_1 (alternative hypothesis) is defined as follows:

There is a statistically significant difference between the model without the independent variables and the model with the independent variables in the prediction of the event "*Author_X* is the author". In other words, there is a significant difference in the influence of the analysed quantitative graph properties for *Author_X* and for *Author_Y*.

[27]In order to standardize the analysed measures, z-score standardization is used, i.e., the standardized measures have zero-mean; and are measured in standard deviation.

We work with this formulation of H_0 and H_1 for all logistic regression models. We perform logistic regression analysis for each pair of authors (10 tested hypotheses in total). Depending on the analysed pair of authors, we substitute *Author_X* and *Author_Y* for *Author_1* and *Author_2*; *Author_1* and *Author_3* etc.

The significance level for each model is $\alpha = 0.05$ which means that there is 5% maximum chance of rejecting H_0while H_0 is correct. However, we are testing 10 hypotheses, where each testing increases the probability of incorrectly rejecting H_0. We address this issue by applying Bonferroni's correction: the increase is compensated for by dividing the significance level α by the total number of performed tests, i.e., each hypothesis is tested at the significance level $\alpha' = 0.05/10 = 0.005$. The results of Chi-Squared Test χ^2 are always right-tailed. In order to calculate logistic regression, we use the programming language R[28].

For each model, we present the results of the Chi-Squared Test χ^2 and the p-value, and we state whether the model with the independent variables provides a better fit than the model without the independent variables.

For each model we also present a coefficients table that includes the calculated values for β_0 (intercept), and the regression coefficients β_1, β_2, β_3 that are inserted into the regression equation, and their confidence intervals. The table also shows p-values that indicate whether the influence of the variable is statistically significant, and odds ratio with its confidence intervals. In order to test the significance of the influence of the individual variables, we apply Bonferroni' s correction to the significance level α by dividing it by the number of regression coefficients, i.e., $\alpha' = 0.05 / 3 = 0.0167$ (for the same reason, the confidence interval of the coefficients is $C.I. = 1 - \alpha' = 0.9833 \approx 98.4\%$).

3.8 Interpretation of Logistic Regression Analysis Results

In the following section we test for each pair of authors whether we can predict the authorship based on the selected quantitative network features. For each pair of authors, we list the results of logistic regression analysis, and we offer the interpretation of these.

[28] "R: The R Project for Statistical Computing." We supply the code for the analysis in R in *Appendix J.*

3.8.1 Prediction of authorship of novels written by Author_1 & Author_2

The independent variables are the standardized selected network features extracted from the novels written by *Author_1* and *Author_2*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_1*; it is assigned value "0", if the values are extracted from the novels written by *Author_2*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_1* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 24.99$, p-value $\approx$ 0. Since the p-value $< \alpha'(0.005)$, H_0 is rejected. The results of logistic regression show that the model is significant; the model with the independent variables provides a better fit than the baseline model without the independent variables. The coefficients table (*Table 17*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)		Chisq	p-value	O.R.	98, 4% C.I. (O.R.)	
Avg. Clustering Coefficient	-4.075	(-10.438,	-1.012)	13.201	0	0.017	(0,	0.364)
Avg. Betweenness Centrality	2.797	(0.618,	6.564)	10.375	0.001	16.4	(1.854,	709.217)
Avg. Closeness Centrality	0.891	(-1.776,	3.754)	0.748	0.387	2.439	(0.169,	42.687)
Intercept	0.653							

***Table 17**: Logistic Regression Analysis Results (Author_1 & Author_2).*

To estimate the probability of the event "*Author_*1 is the author", the value of average clustering coefficient is inserted as x_1, the value of average betweenness centrality is inserted as x_2, and the value of average closeness centrality is inserted as x_3 into formula (2).

$$P = \frac{1}{1 + e^{-(0.653 - 4.075x_1 + 2.797x_2 + 0.891x_3)}} \tag{2}$$

The coefficient β_1 is negative ($\beta_1 = -4.075$); a higher value of average clustering coefficient indicates a lower probability of the event "*Author_1* is the author". Since the p-value (p-value $\approx$ 0) $< \alpha'(0.0167)$ the influence of the given variable is statistically significant. The odds ratio ($O.R. =$ 0.017) indicates that the increase of the given variable by one unit of standard deviation will decrease the odds of the outcome "*Author_1* is the author" by 98.3%.

The coefficient β_2 is positive ($\beta_2 =$ 2.797); a higher value of average betweenness centrality indicates a higher probability of the event "*Author_1* is the author". Since p-value (p-value $= 0.001) < \alpha'(0.0167)$, the influence of the given variable is statistically significant. The odds ratio (O.R. = 16.4) indicates that the increase of the given variable by one unit of standard deviation will increase the odds of the outcome "*Author_1* is the author" by 16.4 times.

The coefficient β_3 is positive ($\beta_3 = 0.891$); a higher value of average closeness centrality indicates a higher probability of the event "*Author_1* is the author". Since p-value (p-value $= 0.387) > \alpha'(0.0167)$, the influence of the given variable is not statistically significant. The odds ratio (O.R. $=$ 2.439) indicates that the increase of the given variable by one unit of standard deviation will increase the odds of the outcome "*Author_1* is the author" by 2.439 times.

The accuracy of prediction of this model is 93.33%. To sum up, the model that predicts whether "*Author_1* is the author" or "*Author_1* is not the author" (i.e., "*Author_2* is the author") based on the independent variables is statistically significant. Two variables are statistically significant: average clustering coefficient and average betweenness centrality.

3.8.2 Prediction of authorship of novels written by Author_1 & Author_3

The independent variables are the standardized selected network features extracted from the novels written by *Author_1* and *Author_3*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_1*; it is assigned value "0", if the values are extracted from the novels written by *Author_3*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_1* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 15.481$, p-value $= 0.001$. Since the p-value $< \alpha'(0.005)$, H_0 is rejected. The results of logistic regression show that the model is significant; the model with the independent variables provides a better fit than the baseline model without the independent variables. The coefficients table (*Table 18*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)		Chisq	p-value	O.R.	98, 4% C.I. (O.R.)	
Avg. Clustering Coefficient	-1.718	(-4.156,	-0.031)	5.954	0.015	0.179	(0.016,	0.97)
Avg. Betweenness Centrality	-1.338	(-3.709,	0.095)	4.848	0.028	0.262	(0.025,	1.1)
Avg. Closeness Centrality	-0.226	(-2.25,	1.511)	0.097	0.756	0.798	(0.105,	4.531)
Intercept	-0.157							

Table 18*: Logistic Regression Analysis Results (Author_1 & Author_3).*

To estimate the probability of the event "*Author_*1 is the author", the value of average clustering coefficient is inserted as x_1, the value of average betweenness centrality is inserted as x_2, and the value of average closeness centrality is inserted as x_3 into the following formula (3).

$$P = \frac{1}{1 + e^{-(-0.157 - 1.718x_1 - 1.338x_2 - 0.226x_3)}} \quad (3)$$

The coefficient β_1 is negative ($\beta_1 = -1.718$); a higher value of average clustering coefficient indicates a lower probability of the event "*Author_1* is the author". Since the p-value (p-value $= 0.015$) $< \alpha'(0.0167)$, the influence of the given variable is statistically significant. The odds ratio ($O.R. =$ 0.179) indicates that the increase of the given variable by one unit of standard deviation will decrease the odds of the outcome "*Author_1* is the author" by 82.1%.

The coefficient β_2 is negative ($\beta_2 = -1.338$); a higher value of average betweenness centrality indicates a lower probability of the event "*Author_1* is the author". Since p-value (p-value $=$ 0.028) $> \alpha'(0.0167)$, the influence of the given variable is not statistically significant. The odds ratio (O.R. $=$ 0.262) indicates that the increase of the given variable by one unit of standard deviation will decrease the odds of the outcome "*Author_1* is the author" by 73.8%.

The coefficient β_3 is negative ($\beta_3 = -0.226$); a higher value of average closeness centrality indicates a lower probability of the event "*Author_1* is the author". Since the p-value (p-value $=$ 0.756) $> \alpha'(0.0167)$, the influence of the given variable is not statistically significant. The odds ratio (O.R. $=$ 0.798) indicates that the increase of the given variable by one unit of standard deviation will increase the odds of the outcome "*Author_1* is the author" by 20.2.%

The accuracy of prediction of this model is 80%. To sum up, the model that predicts whether "*Author_1* is the author" or "*Author_1* is not the author" (i.e., "*Author_3* is the author") based on the independent variables is statistically significant. One variable is statistically significant: average clustering coefficient.

3.8.3 Prediction of authorship of novels written by Author_1 & Author_4

The independent variables are the standardized selected network features extracted from the novels written by *Author_1* and *Author_4*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_1*; it is assigned value "0", if the values are extracted from the novels written by *Author_4*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_1* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 7.509$, p-value $= 0.057$. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 19*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)		Chisq	p-value	O.R.	98, 4% C.I. (O.R.)	
Avg. Clustering Coefficient	-0.717	(-2.212,	0.5)	1.943	0.163	0.488	(0.109,	1.648)
Avg. Betweenness Centrality	0.152	(-1.7,	1.895)	0.046	0.83	1.164	(0.183,	6.652)
Avg. Closeness Centrality	1.506	(-0.014,	3.574)	5.613	0.018	4.507	(0.987,	35.669)
Intercept	-0.064							

***Table 19**: Logistic Regression Analysis Results (Author_1 & Author_4).*

To sum up, the model that predicts whether "*Author_1* is the author" or "*Author_1* is not the author" (i.e., "*Author_4* is the author") based on the independent variables is not statistically significant.

3.8.4 Prediction of authorship of novels written by Author_1 & Author_5

The independent variables are the standardized selected network features extracted from the novels written by *Author_1* and *Author_5*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_1*; it is assigned value "0", if the values are extracted from the novels written by *Author_5*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_1* is the author". The results of Chi-Squared

Test are $\chi^2_{(3)} = 6.604$, p-value $= 0.086$. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 20*) is presented below-

	Coeff.	98, 4% C.I. (Coeff.)	Chisq	p-value	O.R.	98, 4% C.I. (O.R.)
Avg. Clustering Coefficient	-0.721	(-2.009, 0.375)	2.444	0.118	0.486	(0.134, 1.455)
Avg. Betweenness Centrality	0.497	(-0.549, 1.76)	1.247	0.264	1.644	(0.578, 5.812)
Avg. Closeness Centrality	-0.405	(-1.698, 0.694)	0.767	0.381	0.667	(0.183, 2.001)
Intercept	-0.009					

Table 20*: Logistic Regression Analysis Results (Author_1 & Author_5).*

To sum up, the model that predicts whether "*Author_1* is the author" or "*Author_1* is not the author" (i.e., "*Author_5* is the author") based on the independent variables is not statistically significant.

3.8.5 Prediction of authorship of novels written by Author_2 & Author_3

The independent variables are the standardized selected network features extracted from the novels written by *Author_2* and *Author_3*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_2*; it is assigned value "0", if the values are extracted from the novels written by *Author_3*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_2* is the author". Results of Chi-Squared Test are $\chi^2_{(3)} = 8.446$, p-value $= 0.038$. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 21*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)	Chisq	p-value	O.R.	98, 4% C.I. (O.R.)
Avg. Clustering Coefficient	0.056	(-2.339, 2.63)	0.003	0.955	1.058	(0.096, 13.868)
Avg. Betweenness Centrality	-1.309	(-3.656, 0.239)	3.886	0.049	0.27	(0.026, 1.27)
Avg. Closeness Centrality	-0.247	(-3.135, 2.533)	0.052	0.82	0.781	(0.043, 12.594)
Intercept	-0.127					

Table 21*: Logistic Regression Analysis Results (Author_2 & Author_3).*

To sum up, the model that predicts whether "*Author_2* is the author" or "*Author_2* is not the author" (i.e., "*Author_3* is the author") based on the independent variables is not statistically significant.

3.8.6 Prediction of authorship of novels written by Author_2 & Author_4

The independent variables are the standardized selected network features extracted from the novels written by *Author_2* and *Author_4*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_2*; it is assigned value "0", if the values are extracted from the novels written by *Author_4*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_2* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 11.021$, p-value $= 0.012$. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 22*) is presented below.

	Coeff.	**98, 4% C.I. (Coeff.)**		**Chisq**	**p-value**	**O.R.**	**98, 4% C.I. (O.R.)**	
Avg. Clustering Coefficient	1.751	-0.735	5.085	2.676	0.102	5.761	0.48	161.646
Avg. Betweenness Centrality	-2.393	-7.801	0.177	4.477	0.034	0.091	0	1.194
Avg. Closeness Centrality	-0.029	-2.578	2.654	0.001	0.977	0.971	0.076	14.217
Intercept	-0.812							

Table 22 *Logistic Regression Analysis Results (Author_2 & Author_4).*

To sum up, the model that predicts whether "*Author_2* is the author" or "*Author_2* is not the author" (i.e., "*Author_4* is the author") based on the independent variables is not statistically significant.

3.8.7 Prediction of authorship of novels written by Author_2 & Author_5

The independent variables are the standardized selected network features extracted from the novels written by *Author_2* and *Author_5*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_2*; it is assigned value "0", if the values are extracted from the novels written by *Author_5* H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_2* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 6.38$, p-value $=$ 0.095. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 23*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)		Chisq	p-value	O.R.	98, 4% C.I. (O.R.)	
Avg. Clustering Coefficient	1.342	(-0.324,	3.602)	3.588	0.058	3.827	(0.723	36.659)
Avg. Betweenness Centrality	-0.669	(-2.648,	0.983)	0.902	0.342	0.512	(0.071	2.672)
Avg. Closeness Centrality	-0.678	(-3.158,	1.545)	0.533	0.465	0.508	(0.043	4.689)
Intercept	-0.072							

Table 23*: Logistic Regression Analysis Results (Author_2 & Author_5).*

To sum up, the model that predicts whether "*Author_2* is the author" or "*Author_2* is not the author" (i.e., "*Author_5* is the author") based on the independent variables is not statistically significant.

3.8.8 Prediction of authorship of novels written by Author_3 & Author_4

The independent variables are the standardized selected network features extracted from the novels written by *Author_3* and *Author_4*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_3*; it is assigned value "0", if the values are extracted from the novels written by *Author_4*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_3* is the author". Results of Chi-Squared Test are $\chi^2_{(3)} = 15.131$, p-value $= 0.002$. Since the p-value $< \alpha'(0.005)$, H_0 is rejected. The results of logistic regression show that the model is significant; the model with the independent variables provides a better fit than the baseline model without the independent variables. The coefficients table (*Table 24*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)	Chisq	p-value	O.R.	98, 4% C.I. (O.R.)
Avg. Clustering Coefficient	-0.165	(-3.071, 2.359)	0.022	0.881	0.847	(0.046, 10.58)
Avg. Betweenness Centrality	0.915	(-0.721, 2.895)	1.822	0.177	2.497	(0.486, 18.075)
Avg. Closeness Centrality	2.575	(0.048, 7.373)	6.033	0.014	13.135	(1.049, 1592.427)
Intercept	0.276					

***Table 24**: Logistic Regression Analysis Results (Author_3 & Author_4).*

To estimate the probability of the event "*Author_*3 is the author", the value of average clustering coefficient is inserted as x_1, the value of average betweenness centrality is inserted as x_2, and the value of average closeness centrality is inserted as x_3 into formula (4).

$$P = \frac{1}{1 + e^{-(0.276 - 0.165x_1 + 0.915x_2 + 2.575x_3)}} \quad (4)$$

The coefficient β_1 is negative ($\beta_1 = -0.165$); a higher value of average clustering coefficient indicates a lower probability of the event "*Author_3* is the author". Since the p-value (p-value $= 0.881$) $> \alpha'(0.0167)$ the influence of the given variable is not statistically significant. The odds ratio ($O.R. = 0.847$) indicates that the increase of the given variable by one unit of standard deviation will decrease the odds of the outcome "*Author_3* is the author" by 15.3%.

The coefficient β_2 is positive ($\beta_2 = 0.915$); a higher value of average betweenness centrality indicates a higher probability of the event "*Author_3* is the author". Since the p-value (p-value $= 0.177$) $> \alpha'(0.0167)$, the influence of the given variable is not statistically significant. The odds ratio (O.R. $=$ 2.497) indicates that the increase of the given variable by one unit of standard deviation will increase the odds of the outcome "*Author_3* is the author" by 2.497 times.

The coefficient β_3 is positive ($\beta_3 = 2.575$); a higher value of average closeness centrality indicates a higher probability of the event "*Author_3* is the author". Since the p-value (p-value $= 0.014$) $< \alpha'(0.0167)$, the influence of the given variable is statistically significant. The odds ratio (O.R. $= 13.135$) indicates that the increase of the given variable by one unit of standard deviation will increase the odds of the outcome "*Author_3* is the author" by 13.135 times.

The accuracy of prediction of this model is 83.333%. To sum up, the model that predicts whether "*Author_3* is the author" or "*Author_3* is not the author" (i.e., "*Author_4* is the author") based on the independent variables is statistically significant. One variable is statistically significant: average closeness centrality.

3.8.9 Prediction of authorship of novels written by Author_3 & Author_5

The independent variables are the standardized selected network features extracted from the novels written by *Author_3* and *Author_5*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_3*; it is assigned value "0", if the values are extracted from the novels written by *Author_5*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_3* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 9.493$, p-value $= 0.023$. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 25)* is presented below.

	Coeff.	**98, 4% C.I. (Coeff.)**	**Chisq**	**p-value**	**O.R.**	**98, 4% C.I. (O.R.)**
Avg. Clustering Coefficient	1.242	(-0.503, 3.41)	2.831	0.092	3.464	(0.605, 30.262)
Avg. Betweenness Centrality	1.577	(0.077, 3.931)	6.502	0.011	4.842	(1.08, 50.979)
Avg. Closeness Centrality	-0.325	(-2.121, 1.402)	0.223	0.637	0.723	(0.12, 4.063)
Intercept	0.106					

Table 25*: Logistic Regression Analysis Results (Author_3 & Author_5).*

As described above, this model is not statistically significant. However, the results in the coefficients table indicate that there is one independent variable that has significant influence on the dependent variable: average betweenness centrality with the p-value (p-value $= 0.011$) $< \alpha'(0.0167)$. Its regression coefficient β_2 is positive ($\beta_2 = 1.577$); a higher value of average betweenness centrality indicates a higher probability of the event "*Author_3* is the author". The odds ratio (O.R.$=$ 4.842) indicates that the increase of the given variable by one unit of standard deviation will increase the odds of the outcome "*Author_3* is the author" by 4.842 times.

To sum up, the model that predicts whether "*Author_3* is the author" or "*Author_3* is not the author" (i.e., "*Author_5* is the author") based on the independent variables is not statistically significant. However, one variable is statistically significant: average betweenness centrality.

3.8.10 Prediction of authorship of novels written by Author_4 & Author_5

The independent variables are the standardized selected network features extracted from the novels written by *Author_4* and *Author_5*.

The dependent variable is assigned value "1", if the independent values are extracted from novels written by *Author_4*; it is assigned value "0", if the values are extracted from the novels written by *Author_5*. H_0 and H_1 are formulated according to their definition above.

We perform logistic regression analysis to study the influence of independent variables x_1, x_2, and x_3 on the event "*Author_4* is the author". The results of Chi-Squared Test are $\chi^2_{(3)} = 9.357$, p-value $= 0.025$. Since the p-value $> \alpha'(0.005)$, H_0 cannot be rejected. The results of logistic regression show that the model is not significant; the model with the independent variables does not provide a better fit than the baseline model without the independent variables. The coefficients table (*Table 26*) is presented below.

	Coeff.	98, 4% C.I. (Coeff.)	Chisq	p-value	O.R.	98, 4% C.I. (O.R.)
Avg. Clustering Coefficient	0.431	(-1.112, 2.168)	0.453	0.501	1.539	(0.329, 8.737)
Avg. Betweenness Centrality	0.555	(-1.221, 3.069)	0.462	0.496	1.741	(0.295, 21.511)
Avg. Closeness Centrality	-1.813	(-4.505, -0.075)	6.319	0.012	0.163	(0.011, 0.927)
Intercept	0.209					

***Table 26**: Logistic Regression Analysis Results (Author_4 & Author_5).*

As described above, this model is not statistically significant. However, the results in the coefficients table indicate that there is one independent variable that has significant influence on the dependent variable: average closeness centrality with the p-value (p-value $= 0.012$) $< \alpha'(0.0167)$. Its regression coefficient β_3 is negative ($\beta_3 = -1.813$); a higher value of average closeness centrality indicates a lower probability of the event "*Author_4* is the author". The odds ratio (O.R.$= 0.163$) indicates that the increase of the given variable by one unit of standard deviation will decrease the odds of the outcome "*Author_4* is the author" by 83.7%.

To sum up, the model that predicts whether "*Author_4* is the author" or "*Author_4* is not the author" (i.e., "*Author_5* is the author") based on the independent variables is not statistically significant. However, one variable is statistically significant: average closeness centrality.

4 Results

This pen-ultimate chapter presents a summary of the results of the analysis conducted during the *Practical Part.* First, we sum up the obtained results, and we reflect about the qualitative assumptions we can infer from our observations. Second, we reflect about the methodological choices we have made during the practical part and the consequences of their application. We also suggest the direction for future works stemming out of this thesis.

4.1 Summary of Results & Reflection

First, we briefly sum up the results of the binary logistic regression analysis. We have tested 10 hypotheses, i.e., for each pair of analysed authors we have explored whether it is possible to estimate the authorship of the novels based on their average clustering coefficient, average betweenness centrality and average closeness centrality.

Out of the 10 tested hypotheses, the results of logistic regression analysis indicated that the model with the selected features provided a better fit than the model without the features in exactly 3 cases (*Author_1 & Author_2*; *Author_1 & Author_3; Author_3 & Author_4)*; with average clustering coefficient and average betweenness centrality showing significant influence in the first case, average clustering coefficient in the second case, and average closeness centrality in the third case. The model predicting the authorship of *Author_3 & Author_5*, and the model predicting the authorship of *Author_4 & Author_5*, showed no statistical significance as a whole. However, the results indicated that the influence of average betweenness centrality was significant in the former case, and the closeness centrality in the latter. The remaining 5 models did not provide a better fit than the baseline model, nor the influence of any feature was statistically significant.

We present the summary of the results in *Table 27*. The table shows for each model whether it was statistically significant as a whole. It also provides information about the accuracy of the prediction and highlights the significant network features for the respective models.

	Significant	Accuracy (%)	Significant Features
Author_1 & Author_2	YES	93.333	Average clustering coefficient, Average betweenness centrality
Author_1 & Author_3	YES	80	Average clustering coefficient
Author_1 & Author_4	NO	73.333	-
Author_1 & Author_5	NO	70	-
Author_2 & Author_3	NO	73.333	-
Author_2 & Author_4	NO	70	-
Author_2 & Author_5	NO	63.333	-
Author_3 & Author_4	YES	83.333	Average closeness centrality
Author_3 & Author_5	NO	70	Average betweenness centrality
Author_4 & Author_5	NO	63.333	Average closeness centrality

Table 27*: Summary of Logistic Regression Analysis Results.*

Our analysis proved that we can predict in a statistically significant way the authorship of Alger, Horatio Jr. (*Author_1*) as opposed to J. A. Altsheler (*Author_*2) based on the lower value of average clustering coefficient and higher value of average betweenness centrality. Based on the gained knowledge, we can attempt to make a generalization of the character networks built by Alger and Altsheler.

Alger's networks have a lower clustering coefficient, i.e., the neighbouring nodes do not share interactions among themselves as much as they do in the case of Altsheler's networks. The average betweenness centrality is higher in Alger's novels. From that we can infer that his networks need more characters in the position of "bridges" whose removal would disrupt the information flow, or a character via which a lot of the shortest paths pass.

Similarly, we can predict in a statistically significant way the authorship of Alger, Horatio Jr. (*Auhor_1*) as opposed to E. S. Ellis (*Author_3*) based on the lower value of average clustering coefficient. From this observation we can infer that the local density of Alger's networks is lower compared to Ellis' networks, i.e., the characters in Alger's novels do not tend to co-occur with as many characters as they do in Ellis' novels.

Lastly, we can predict in a statistically significant way the authorship of E. S. Ellis (*Auhor_3*) as opposed to G. A. Henty (*Author_4*) based on the higher value of average closeness centrality. Consequently, we can expect the nodes in Ellis' networks to be closer to each other than the nodes in Henty's networks, i.e., the characters in Ellis' novels are on average more "within reach" than the characters in Henty's novels.

We provide examples of networks for the models (combinations of authors) that proved to be significant. Example (1) shows networks built by Alger (*Author_1*) and Altsheler (*Author_2)*. The colour of the nodes reflects their clustering coefficient, the size of the nodes reflects their betweenness centrality. Example (2) shows networks built by Alger (*Author_1*) and Ellis (*Author_3)*. The size of the nodes reflects their degree, the colour reflects their clustering coefficient. Example (3) shows networks built by Ellis (*Author_3)* and Henty (*Author_4*). The size of the nodes and their colour reflects their closeness centrality. The networks presented in the examples were chosen based on high accuracy of the prediction by the respective models.

(1)

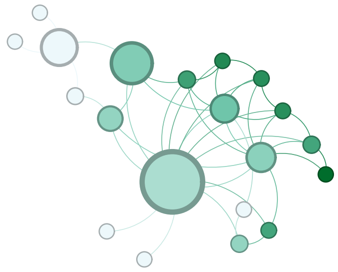


Figure 19: *Author_1, text_1 (short)*

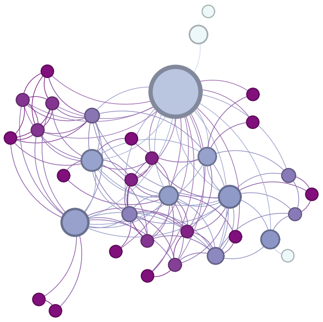


Figure 20: *Author_2, text_11(short)*

(2)

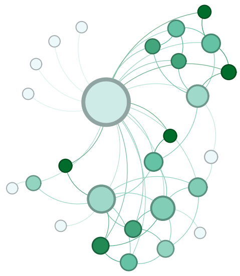


Figure 21: *Author_1, text_5 (short)*

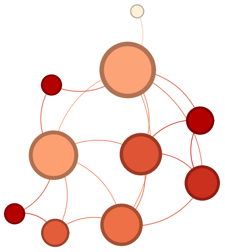


Figure 22: *Author_3, text_12 (short)*

(3)

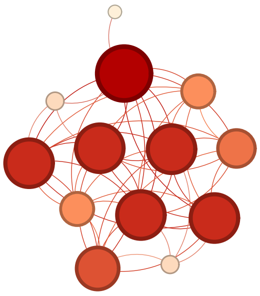


Figure 23: *Author_3, text_15 (short)*

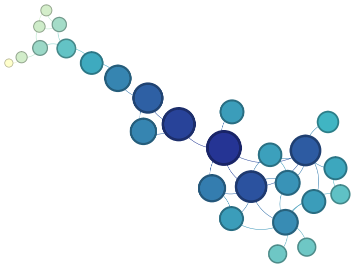


Figure 24: *Author_4, text_1(short)*

The remaining seven tested models did not provide a better fit than the baseline model without the selected variables, i.e., the selected features did not contribute to authorship attribution.

Based on the results we can conclude that individual authors indeed do build their character networks distinctively, and their networks might differ significantly in terms of average clustering coefficient, average betweenness centrality and average closeness centrality. However, it is a matter of specific pairs of authors. For example, in our study, the authorship of Alger or Altsheler can be attributed based on the values of average clustering coefficient and of average betweenness centrality. Nevertheless, that is not the case for other pairs of authors.

We have not identified any specific network feature that would contribute to the authorship attribution task in general. However, the obtained results for individual pairs of authors and individual network features are very interesting and, in some cases, promising. Based on the results we cannot make any generalizations about the significance of the selected network features for unknown pairs of authors. Nonetheless, we believe that the topic of authorship attribution based on quantitative network features is worth further exploration.

4.2 Discussion

The results we have obtained are inherently influenced by the methodological choices we have made during the execution of the *Practical Part*. In this section, we reflect about these choices, their consequences, and we suggest the direction for future research on the topic.

First, the results are influenced by the sample size upon which the analysis was performed. We studied 75 3rd person narratives of the same genre written by five authors. Of course, a larger dataset would result into more precise results.

Another thing to consider about the sample novels is their varying length. We have performed correlation analysis to confirm or discard our presupposition that the values of the network features and the length of the novels are correlated. The results indicated that there is indeed correlation, which we have addressed by shortening the novels to the length of first 50,000 tokens.

Implementing this measure, we have given the authors the same space to develop their networks, but we have ignored the fact that character distribution is likely influenced by the final length of the novel. Therefore, for some of the novels we have extracted almost complete temporal integration networks, whereas for other only partial temporal integration. An alternative to our solution would be chunking the novels, as described earlier.

Second, the extracted values of network features are conditioned by the approach we have adopted towards character network extraction. The process requires a lot of methodological choices. We have described these thoroughly and we have offered our reasoning for adopting our approach. We have used a partially automated approach, using spaCy NER and additional post-processing. We have based interactions on co-occurrences of the characters in the same sentence, and we have decided to perform unification of the characters. We have extracted undirected, unweighted, unsigned and static networks.

What would have the results looked like if we had used spaCy default NER model, without post-processing? If we had based the interactions on conversations or direct actions? Would the networks have reflected intimacy, if we had decided against character unification, and would it have resulted in an interesting authorial feature? Different methodological choices would intrinsically lead to the extraction of different network feature values, and possibly to different results of logistic regression analysis.

Third, we have offered reasoning for extracting the three final features of interest (average clustering coefficient, average betweenness centrality, average closeness centrality). However, we believe it would be also interesting to analyse the influence of other network features (e.g., modularity, average eigenvector centrality).

To sum up, we believe that authorship attribution based on quantitative network features is worth further exploration. For the future works, we recommend a more elaborate data selection, as well as analysing the whole-length novels or chunking. We recommend experimenting with the character network extraction process. We suggest extracting directed, weighted and signed networks, or any combination of these.

We think that extracting dynamic networks would be of special interest to authorship attribution, as it would enable us to study how the authors build their networks and how the characters' interactions evolve over time, e.g., chapter by chapter. Finally, we recommend studying the influence of different quantitative network features than the ones selected for our analysis.

We have made the methodological choices in line with the aim and scope of this thesis, and we have presented reasoning for doing so. Nevertheless, there remains a lot to be uncovered and explored in the relationship between quantitative character network features and the authorship attribution task.

5 Conclusion

The aim of this Bachelor's Thesis was to explore whether quantitative character network features contribute to the authorship attribution task, i.e., whether the way an author builds the network of characters' interactions is an authorial feature.

In order to answer this question, we first needed to gain a better understanding of Science Network, Graph Theory, and Social Network Analysis, which we introduce in the *Theoretical Part*. This part also provides an introduction to character networks and a description of different authors' approach towards the task of character network extraction.

In the *Practical Part* of the thesis, we first describe the data that enter the analysis. We analyse 75 novels of children fiction written by five distinct authors, i.e., 15 novels per author. Then we provide a thorough description of the approach we have adopted towards character network extraction. We have followed a partially automated approach and used spaCy NER with additional post-processing to extract a character network for each of the analysed novels. The resulting networks are undirected, unweighted, unsigned, static and the interactions are based on co-occurrences of the characters.

Subsequently we extract the quantitative network features of interest, and we perform correlation analysis to see whether the features and the length of the novels correlate. As we conclude that indeed they do, we shorten the novels to the length of first 50, 000 tokens and we work with the values of the features of the shortened novels. Using Exploratory Factor Analysis, we narrow down the number of network features of interest to three: average clustering coefficient, average betweenness centrality and average closeness centrality. Then we use Binary Logistic Regression to predict the authorship of each pair of authors based on the selected network features, and we interpret the results.

Finally, we sum up the results and we reflect about the qualitative assumption we can infer from these. Out of the 10 tested hypotheses, three models with the selected features provided a better fit than the baseline model without the features. In the first case average clustering coefficient and average betweenness centrality showed significant influence, in the second case only average clustering coefficient showed significant influence, and in the third case average closeness centrality showed significant influence on authorship prediction. In two cases the models were not statistically significant as a whole, however the influence of average betweenness centrality in one case and of a of average closeness centrality in the other case showed statistical significance.

We conclude that for some pairs of authors the authorship can be attributed based on selected network features, but we cannot draw general conclusions about unknown pairs of authors, nor can we state that any given feature contributes to authorship attribution in general. In the last part of the thesis, we reflect about the methodological choices we have made, and we suggest further research on the topic, adopting different approaches. We believe that authorship attribution based on quantitative network features has more to offer, and we recommend its further exploration.

6 Bibliography

1. References

Agarwal, Apoorv, Augusto Corvalan, Jacob A. Jensen, and Owen Rambow. 2012. "Social Network Analysis of Alice in Wonderland." In *Proceedings of the NAACL-HLT 2012 Workshop on computational linguistics for literature*: 88-96.

Akoglu, Haldun. 2018. “User’s Guide to Correlation Coefficients.” *Turkish Journal of Emergency Medicine* 18 (3): 91–93. https://doi.org/10.1016/j.tjem.2018.08.001.

Ardanuy, Mariona Coll, and Caroline Sporleder. 2014. “Structure-Based Clustering of Novels.” *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL)*: 31-39. https://doi.org/10.3115/v1/w14-0905.

———. 2015. “Clustering of Novels Represented as Social Networks.” *Linguistic Issues in Language Technology* 12 (October). https://doi.org/10.33011/lilt.v12i.1379.

Barabási, Albert-László. n.d. “Network Science by Albert-László Barabási.” BarabásiLab. Accessed June 8, 2023. http://networksciencebook.com.

———. 2012. eBook *Network Science* (November 2012).

Blondel, Vincent D., Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. “Fast Unfolding of Communities in Large Networks.” *Journal of Statistical Mechanics: Theory and Experiment* 2008 (10): P10008. https://doi.org/10.1088/1742-5468/2008/10/p10008.

Bossaert, Goele, and Nadine Meidert. 2013. “‘We Are Only as Strong as We Are United, as Weak as We Are Divided’ a Dynamic Analysis of the Peer Support Networks in the Harry Potter Books.” *Open Journal of Applied Sciences* 03 (02): 174–85.

Chen, Bikun, and Yuefen Wang. 2016. “Character Interaction Network Analysis of Chinese Literary Work- A Preliminary Study.” *Proceedings of the Association for Information Science and Technology* 53 (1): 1–4. https://doi.org/10.1002/pra2.2016.14505301088.

Das, Kousik, Sovan Samanta, and Madhumangal Pal. 2018. “Study on Centrality Measures in Social Networks: A Survey.” *Social Network Analysis and Mining* 8 (1). https://doi.org/10.1007/s13278-018-0493-2.

DeCoster, Jamie. 1998. “Overview of Factor Analysis.” http://www.stat-help.com/factor.pdf.

Dekker, Niels, Tobias Kuhn, and Marieke Van Erp. 2019. “Evaluating Named Entity Recognition Tools for Extracting Social Networks from Novels.” *PeerJ* 5 (April): e189. https://doi.org/10.7717/peerj-cs.189.

DeMaris, Alfred. 1995. “A Tutorial in Logistic Regression.” *Journal of Marriage and Family* 57 (4): 956. https://doi.org/10.2307/353415.

Elsner, Micha. 2012. "Character-Based Kernels for Novelistic Plot Structure." *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*: 634-644.https://www.aclweb.org/anthology/E12-1065.pdf.

Elson, David K. 2012. *Modeling narrative discourse*. Columbia University.

Elson, David K., Kathleen R. McKeown, and Nicholas Dames. 2010. "Extracting Social Networks from Literary Fiction." https://doi.org/10.7916/d8m90j1s.

GrammarBook.com. 2021. "Capitalizing Titles - The Blue Book of Grammar and Punctuation." *The Blue Book of Grammar and Punctuation*, June. https://www.grammarbook.com/blog/capitalization/capitalizing-titles/.

Grener, Adam, Markus Luczak-Roesch, Emma Fenton, and Thomas Goldfinch. 2017. "Towards a Computational Literary Science: A Computational Approach to Dickens' Dynamic Character Networks." *Technical Report.*, January. https://doi.org/10.5281/zenodo.259499.

Hutchinson, Sterling, Vivek V. Datla, and Max M. Louwerse. 2012. "Social Networks Are Encoded in Language." *Cognitive Science* 34 (34). https://escholarship.org/content/qt3d88c1j1/qt3d88c1j1.pdf.

Labatut, Vincent, and Xavier Bost. 2019. "Extraction and Analysis of Fictional Character Networks." *ACM Computing Surveys* 52 (5): 1–40. https://doi.org/10.1145/3344548.

LaValley, Michael P. 2008. "Logistic Regression." *Circulation* 117 (18): 2395–99. https://doi.org/10.1161/circulationaha.106.682658.

Lee, John D., and Chak Yan Yeung. 2012. "Extracting Networks of People and Places from Literary Texts." In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*: 209-218. https://aclanthology.org/Y12-1022.pdf

Moretti, Franco. 2011. "Network theory, plot analysis. Literary Lab Pamphlet 2." https://litlab.stanford.edu/LiteraryLabPamphlet2.pdf

Nadeau, David R., and Satoshi Sekine. 2007. "A Survey of Named Entity Recognition and Classification." *Lingvisticae Investigationes* 30 (1): 3–26.

Prion, Susan, and Katie Anne Haerling. 2014. "Making Sense of Methods and Measurement: Spearman-Rho Ranked-Order Correlation Coefficient." *Clinical Simulation in Nursing* 10 (10): 535–36. https://doi.org/10.1016/j.ecns.2014.07.005.

Rochat, Yannick, and Frédéric Kaplan. 2014. "Analyse Des Réseaux de Personnages Dans Les Confessions de Jean-Jacques Rousseau." *Les Cahiers Du Numérique* 10 (3): 109–33. https://doi.org/10.3166/lcn.10.3.109-133.

Sack, Graham Alexander. 2012. "Character Networks for Narrative Generation." *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 8 (2): 38–43. https://doi.org/10.1609/aiide.v8i2.12541.

Seo, Jong-Kyu, Sunghwan Kim, Haesung Tak, and Hwan-Gue Cho. 2014. “A Structural Analysis of Literary Fictions with Social Network Framework.” In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*:634-640. https://doi.org/10.1145/2554850.2555049

spaCy. n.d.-a. “Rule-Based Matching.” spaCy Usage Documentation. Accessed March 6, 2023. https://spacy.io/usage/rule-based-matching.

———. n.d.-b. “SpaCy 101: Everything You Need to Know.” spaCy Usage Documentation. Accessed March 6, 2023. https://spacy.io/usage/spacy-101.

Sperandei, Sandro. 2014. “Understanding Logistic Regression Analysis.” *Biochemia Medica* 24(1):*12–8*. https://doi.org/10.11613/bm.2014.003.

Stamatatos, Efstathios. 2009. “A Survey of Modern Authorship Attribution Methods.” *Journal of the Association for Information Science and Technology* 60 (3): 538–56. https://doi.org/10.1002/asi.21001.

Stoltzfus, Jill. 2011. “Logistic Regression: A Brief Primer.” *Academic Emergency Medicine* 18 (10): 1099–1104. https://doi.org/10.1111/j.1553-2712.2011.01185.x.

Trovati, Marcello, and James Brady. 2014. *Towards an Automated Approach to Extract and Compare Fictional Networks: An Initial Evaluation*. https://doi.org/10.1109/dexa.2014.58.

Tsvetovat, Maksim, and Alexander Kouznetsov. 2011. *Social Network Analysis for Startups: Finding Connections on the Social Web*. “O’Reilly Media, Inc.”

University of Sussex. n.d. “Abbreviations : Capital Letters and Abbreviations.”Acessed April 4, 2023. https://www.sussex.ac.uk/informatics/punctuation/capsandabbr/abbr.

Vala, Hardik, David Jurgens, Andrew Piper, and Derek Ruths. 2015. “*Mr. Bennet, His Coachman, and the Archbishop Walk into a Bar but Only One of Them Gets Recognized: On The Difficulty of Detecting Characters in Literary Texts*.” In In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*: 769-774. https://doi.org/10.18653/v1/d15-1088.

Van Dalen-Oskam, K.H., Jesse De Does, Maarten Marx, Isaac Sijaranamual, Katrien Depuydt, Boukje Verheij, and Valentijn Geirnaert. 2014. *Named Entity Recognition and Resolution for Literary Studies. Computational Linguistics in the Netherlands*. Vol. 4. https://pure.knaw.nl/ws/files/798059/2014_VanDalenOskam_07_Namescape.pdf.

Vittinghoff, Eric, and Charles E. McCulloch. 2007. “Relaxing the Rule of Ten Events per Variable in Logistic and Cox Regression.” *American Journal of Epidemiology* 165 (6): 710–18. https://doi.org/10.1093/aje/kwk052.

Waumans, Michael, Thibaut Nicodème, and Hugues Bersini. 2015. “Topology Analysis of Social Networks Extracted from Literature.” *PLOS ONE* 10 (6): e0126470. https://doi.org/10.1371/journal.pone.0126470.

Yong, An Gie, and Sean. Pearce. 2013. "A Beginner's Guide to Factor Analysis: Focusing on Exploratory Factor Analysis." *Tutorials in Quantitative Methods for Psychology* 9 (2): 79–94. https://doi.org/10.20982/tqmp.09.2.p079.

Zweig, Katharina A. 2016. *Network Analysis Literacy: A Practical Approach to the Analysis of Networks*. Springer.

2. Footnotes

Barabási, Albert-László. "Network Science by Albert-László Barabási." BarabásiLab. Accessed June 8, 2023. http://networksciencebook.com/chapter/2#bridges

———. "Network Science by Albert-László Barabási." BarabásiLab. Accessed June 8, 2023. http://networksciencebook.com

"Determiners (the, My, Some, This)." In *Cambridge Grammar*. Accessed April 4, 2023. https://dictionary.cambridge.org/grammar/british-grammar/determiners-the-my-some-this.

Fletcher, Joseph Smith *Scarhaven Keep*. Urbana, Illinois: Project Gutenberg, 2006.

Gephi. "Gephi - The Open Graph Viz Platform." Accessed April 10, 2023. https://gephi.org/.

Nameparser 1.1.2 Documentation. "HumanName Class Documentation." Accessed April 4, 2023. https://nameparser.readthedocs.io/en/latest/modules.html#module-nameparser.parser.

Nameparser 1.1.2 Documentation. "Python Human Name Parser." Accessed April 4, 2023. https://nameparser.readthedocs.io/en/latest/index.html#.

NetworkX Documentation. "NetworkX." Accessed April 10, 2023. https://networkx.org/.

NetworkX 3.1 Documentation. "Betweenness_centrality." Accessed April 10, 2023. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.betweenness_centrality.html.

NetworkX 3.1 Documentation. "Closeness_centrality." Accessed April 10, 2023. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.closeness_centrality.html.

NetworkX 3.1 Documentation. "Degree_centrality." Accessed April 10, 2023. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.degree_centrality.html.

NetworkX 3.1 Documentation. "Eigenvector_centrality." Accessed April 10, 2023. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.eigenvector_centrality.html.

NetworkX 3.1 Documentation. "Louvain_communities." Accessed June 8, 2023. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.louvain.louvain_communities.html.

Project Gutenberg. "Project Gutenberg." Accessed April 3, 2023. https://www.gutenberg.org/.

Python. "Welcome to Python.Org." Accessed June 7, 2023. https://www.python.org/.

Quita. "QUITA Online." Accessed April 26, 2023. https://kol.ff.upol.cz/quita/.

R Project. "R: The R Project for Statistical Computing." Accessed April 26, 2023. https://www.r-project.org/.

Rowling, J. K. *Harry Potter and the Philosopher's Stone*. Bloomsbury Publishing, 2014.

"Royalty, Aristocracy & Titles - SMART Vocabulary Cloud with Related Words and Phrases." In *Cambridge Dictionary*. Accessed April 5, 2023. https://dictionary.cambridge.org/topics/society/royalty-aristocracy-and-titles/.

Social Security Administration(.gov). “Popular Baby Names.” Accessed April 5, 2023. https://www.ssa.gov/cgi-bin/popularnames.cgi.

spaCy. “spaCy · Industrial-Strength Natural Language Processing in Python.” Accessed June 8, 2023. https://spacy.io/.

Statskingdom.com. “Correlation Coefficient Calculator - Including the Covariance and Calculation Steps.” Accessed April 25 & 29, 2023. https://www.statskingdom.com/correlation-calculator.html.

7 Appendices

Appendix A: List of the analysed novels with the indication of their length in tokens

Author_1, novels by Alger, Horatio, Jr.:		Length
text_1	Alger, Horatio, Jr. Herbert Carter's Legacy; Or, the Inventor's Son. Urbana, Illinois: Project Gutenberg, 2004.	61,687
text_2	Alger, Horatio, Jr. Jack's Ward; Or, The Boy Guardian. Urbana, Illinois: Project Gutenberg, 2004.	62,447
text_3	Alger, Horatio, Jr. The Young Explorer; Or, Claiming His Fortune. Urbana, Illinois: Project Gutenberg, 2004.	54,239
text_4	Alger, Horatio, Jr. Walter Sherwood's Probation. Urbana, Illinois: Project Gutenber, 2004.	60,015
text_5	Alger, Horatio, Jr. The Store Boy. Urbana, Illinois: Project Gutenberg, 2004.	60,406
text_6	Alger, Horatio, Jr. Brave and Bold; Or, The Fortunes of Robert Rushton. Urbana, Illinois: Project Gutenberg, 2006.	70,763
text_7	Alger, Horatio, Jr. Paul Prescott's Charge. Urbana, Illinois: Project Gutenberg, 2006.	75,545
text_8	Alger, Horatio, Jr. Struggling Upward, or Luke Larkin's Luck. Urbana, Illinois: Project Gutenberg, 2004.	67,047
text_9	Alger, Horatio, Jr. Bound to Rise; Or, Up the Ladder. Urbana, Illinois: Project Gutenberg, 2004.	61,749
text_10	Alger, Horatio, Jr. Hector's Inheritance, Or, the Boys of Smith Institute. Urbana, Illinois: Project Gutenberg, 2004.	64,528
text_11	Alger, Horatio, Jr. Try and Trust; Or, Abner Holden's Bound Boy. Urbana, Illinois: Project Gutenberg, 2004.	72,948
text_12	Alger, Horatio, Jr. Sink or Swim; or, Harry Raymond's Resolve. Urbana, Illinois: Project Gutenberg, 2019.	79,404
text_13	Alger, Horatio, Jr. Driven from Home; Or, Carl Crawford's Experience. Urbana, Illinois: Project Gutenberg, 2006.	68,689
text_14	Alger, Horatio, Jr. Helping Himself; Or, Grant Thornton's Ambition. Urbana, Illinois: Project Gutenberg, 2004.	63,641
text_15	Alger, Horatio, Jr. Frank's Campaign; Or, The Farm and the Camp. Urbana, Illinois: Project Gutenberg, 1998.	76,554

Author_2, novels by Altsheler, Joseph A.:		Length
text_1	Altsheler, Joseph A. The Masters of the Peaks: A Story of the Great North Woods. Urbana, Illinois: Project Gutenberg, 2004.	97,497
text_2	Altsheler, Joseph A. The Scouts of the Valley. Urbana, Illinois: Project Gutenberg, 2004.	128,571
text_3	Altsheler, Joseph A. The Shadow of the North: A Story of Old New York and a Lost Campaign. Urbana, Illinois: Project Gutenberg, 2004.	114,091
text_4	Altsheler, Joseph A. The Forest of Swords: A Story of Paris and the Marne. Urbana, Illinois: Project Gutenberg, 2005.	98,398

text_5	Altsheler, Joseph A. The Star of Gettysburg: A Story of Southern High Tide. Urbana, Illinois: Project Gutenberg, 2003.	115,291
text_6	Altsheler, Joseph A. The Hunters of the Hills. Urbana, Illinois: Project Gutenberg, 2005.	112,440
text_7	Altsheler, Joseph A. The Hosts of the Air. Urbana, Illinois: Project Gutenberg, 2005.	99,559
text_8	Altsheler, Joseph A. The Forest Runners: A Story of the Great War Trail in Early Kentucky. Urbana, Illinois: Project Gutenberg, 2005.	95,866
text_9	Altsheler, Joseph A. The Shades of the Wilderness: A Story of Lee's Great Stand. Urbana, Illinois: Project Gutenberg, 2004.	109,415
text_10	Altsheler, Joseph A. The Last of the Chiefs: A Story of the Great Sioux War. Urbana, Illinois: Project Gutenberg, 2007.	110,216
text_11	Altsheler, Joseph A. The Texan Scouts: A Story of the Alamo and Goliad. Urbana, Illinois: Project Gutenberg, 2005.	125,160
text_12	Altsheler, Joseph A. The Guns of Bull Run: A Story of the Civil War's Eve. Urbana, Illinois: Project Gutenberg, 2003.	109,801
text_13	Altsheler, Joseph A. The Young Trailers: A Story of Early Kentucky. Urbana, Illinois: Project Gutenberg, 2006.	86,453
text_14	Altsheler, Joseph A. The Rock of Chickamauga: A Story of the Western Crisis. Urbana, Illinois: Project Gutenberg, 2006.	100,327
text_15	Altsheler, Joseph A. The Sun of Quebec: A Story of a Great Crisis. Urbana, Illinois: Project Gutenberg, 2006.	119,878

Author_3, novels by Ellis, Edward Sylvester:		Length
text_1	Ellis, Edward Sylvester. Adrift in the Wilds; Or, The Adventures of Two Shipwrecked Boys. Urbana, Illinois: Project Gutenberg, 2007.	74,635
text_2	Ellis, Edward Sylvester. Cowmen and Rustlers: A Story of the Wyoming Cattle Ranges. Urbana, Illinois: Project Gutenberg, 2004.	64,353
text_3	Ellis, Edward Sylvester. A Waif of the Mountains. Urbana, Illinois: Project Gutenberg, 2009.	83,586
text_4	Ellis, Edward Sylvester. Two Boys in Wyoming: A Tale of Adventure. Urbana, Illinois: Project Gutenberg, 2006.	83,102
text_5	Ellis, Edward Sylvester. The Life of Kit Carson: Hunter, Trapper, Guide, Indian Agent and Colonel U.S.A. Urbana, Illinois: Project Gutenberg, 2005.	69,168
text_6	Ellis, Edward Sylvester. The Cave in the Mountain. Urbana, Illinois: Project Gutenberg, 2005.	67,202
text_7	Ellis, Edward Sylvester. Brave Tom; Or, The Battle That Won. Urbana, Illinois: Project Gutenberg, 2004	59,739
text_8	Ellis, Edward Sylvester. The Young Scout: The Story of a West Point Lieutenant. Urbana, Illinois: Project Gutenberg, 2018	74,363
text_9	Ellis, Edward Sylvester. The Land of Mystery. Urbana, Illinois: Project Gutenberg, 2005.	72,957
text_10	Ellis, Edward Sylvester. Through Forest and Fire. Urbana, Illinois: Project Gutenberg, 2005	59,268

text_11	Ellis, Edward Sylvester. Adrift on the Pacific: A Boys [sic] Story of the Sea and its Perils. Urbana, Illinois: Project Gutenberg, 2009.	66,394
text_12	Ellis, Edward Sylvester. In the Pecos Country. Urbana, Illinois: Project Gutenberg, 2004.	66,252
text_13	Ellis, Edward Sylvester. The Boy Patrol on Guard. Urbana, Illinois: Project Gutenberg, 2013.	64,796
text_14	Ellis, Edward Sylvester. The Phantom of the River. Urbana, Illinois: Project Gutenberg, 2007.	69,644
text_15	Ellis, Edward Sylvester. Footprints in the Forest. Urbana, Illinois: Project Gutenberg, 2008.	78,730

Author_4, novels by Henty, George Alfred:		Length
text_1	Henty, George Alfred. For Name and Fame; Or, Through Afghan Passes. Urbana, Illinois: Project Gutenberg, 2007.	102,474
text_2	Henty, George Alfred. On the Pampas; Or, The Young Settlers. Urbana, Illinois: Project Gutenberg, 2004.	103,138
text_3	Henty, George Alfred. Through the Fray: A Tale of the Luddite Riots. Urbana, Illinois: Project Gutenberg, 2005.	121,229
text_4	Henty, George Alfred. In Times of Peril: A Tale of India. Urbana, Illinois: Project Gutenberg, 2004.	126,575
text_5	Henty, George Alfred. Orange and Green: A Tale of the Boyne and Limerick. Urbana, Illinois: Project Gutenberg, 2006.	111,480
text_6	Henty, George Alfred. A Final Reckoning: A Tale of Bush Life in Australia. Urbana, Illinois: Project Gutenberg, 2006.	122,179
text_7	Henty, George Alfred. In the Reign of Terror: The Adventures of a Westminster Boy. Urbana, Illinois: Project Gutenberg, 2003.	109,667
text_8	Henty, George Alfred. St. George for England. Urbana, Illinois: Project Gutenberg, 2002.	114,538
text_9	Henty, George Alfred. The Lion of the North: A Tale of the Times of Gustavus Adolphus. Urbana, Illinois: Project Gutenberg, 2004.	121,157
text_10	Henty, George Alfred. On the Irrawaddy: A Story of the First Burmese War. Urbana, Illinois: Project Gutenberg, 2007.	125,987
text_11	Henty, George Alfred. At Aboukir and Acre: A Story of Napoleon's Invasion of Egypt. Urbana, Illinois: Project Gutenberg, 2007.	128,016
text_12	Henty, George Alfred. At Agincourt. Urbana, Illinois: Project Gutenberg, 2004.	139,250
text_13	Henty, George Alfred. Winning His Spurs: A Tale of the Crusades. Urbana, Illinois: Project Gutenberg, 2004.	110,955
text_14	Henty, George Alfred. Jack Archer: A Tale of the Crimea. Urbana, Illinois: Project Gutenberg, 2004.	112,817
text_15	Henty, George Alfred. The Boy Knight: A Tale of the Crusades. Urbana, Illinois: Project Gutenberg, 2004.	111,432

Author_5, novels by Optic, Oliver:		Length
text_1	Optic, Oliver. Taken by the Enemy. Urbana, Illinois: Project Gutenberg, 2006.	74,020
text_2	Optic, Oliver. The Soldier Boy; or, Tom Somers in the Army: A Story of the Great Rebellion. Urbana, Illinois: Project Gutenberg, 2005.	83,965
text_3	Optic, Oliver. The Coming Wave; Or, The Hidden Treasure of High Rock. Urbana, Illinois: Project Gutenberg, 2007.	77,543
text_4	Optic, Oliver. The Yacht Club; or, The Young Boat-Builder. Urbana, Illinois: Project Gutenberg, 2007.	77,539
text_5	Optic, Oliver. Stand By The Union. Urbana, Illinois: Project Gutenberg, 2006.	79,188
text_6	Optic, Oliver. Across India; Or, Live Boys in the Far East. Urbana, Illinois: Project Gutenberg, 2005.	106,834
text_7	Optic, Oliver. On The Blockade. Urbana, Illinois: Project Gutenberg, 2006.	76,674
text_8	Optic, Oliver. Now or Never; Or, The Adventures of Bobby Bright: A Story for Young Folks. Urbana, Illinois: Project Gutenberg, 2005.	55,987
text_9	Optic, Oliver. Four Young Explorers; Or, Sight-Seeing in the Tropics. Urbana, Illinois: Project Gutenberg, 2008.	101,763
text_10	Optic, Oliver. All Aboard; or, Life on the Lake. Urbana, Illinois: Project Gutenberg, 2005.	50,997
text_11	Optic, Oliver. Work and Win; Or, Noddy Newman on a Cruise. Urbana, Illinois: Project Gutenberg, 2007.	59,867
text_12	Optic, Oliver. Poor and Proud; Or, The Fortunes of Katy Redburn: A Story for Young Folks. Urbana, Illinois: Project Gutenberg, 1996.	58,110
text_13	Optic, Oliver. A Victorious Union. Urbana, Illinois: Project Gutenberg, 2006.	78,530
text_14	Optic, Oliver. Haste and Waste; Or, the Young Pilot of Lake Champlain. A Story for Young People.Urbana, Illinois: Project Gutenberg, 2004.	60,922
text_15	Optic, Oliver. Fighting for the Right.Urbana, Illinois: Project Gutenberg, 2006.	78,778

Appendix B: Source code for the character network extraction in Python

```
import text_modul ### funkce read_text_file
import spacy
from spacy.language import Language
import json
import re
from collections import Counter
from nameparser import HumanName
from spacy.lang.en import English
from itertools import combinations
import networkx as nx

text = text_modul.read_text_file(file_name)
text = text.replace("\n\n", " ")
text = text.replace("\n", " ")

with open("titles_and_honorifics_list", "r") as f:
    titles_and_honorifics= json.load(f)
with open("determiners", "r") as f:
    determiners = json.load(f)
with open("titles_capitalized_full_stop", "r") as f:
    titles_with_full_stop = json.load(f)

###modified_default_pipeline
nlp = spacy.load("en_core_web_trf")
@Language.component("remove_non_person")

def remove_non_person(doc):
    original_ents = list(doc.ents)
    for ent in doc.ents:
        if ent.label_ != "PERSON":
            original_ents.remove(ent)
    doc.ents = original_ents
    return(doc)

@Language.component("expand_person_entities")

def expand_person_entities(doc):
    new_ents = []
    for ent in doc.ents:
        if ent.label_ == "PERSON" and ent.start != 0:
            previous_token = doc[ent.start -1]
            if previous_token.text in titles_and_honorifics:
                new_ent = Span(doc, ent.start -1, ent.end, label = ent.label)
                new_ents.append(new_ent)
            else:
                new_ents.append(ent)
        else:
            new_ents.append(ent)
    doc.ents = new_ents
    return doc

nlp.add_pipe("remove_non_person")
nlp.add_pipe("expand_person_entities", after="ner")

doc = nlp(text)

###further post-processing of modified_default_pipeline
characters = []

for ent in doc.ents:
    characters.append(ent.text)

def remove_non_frequent_occurrences(characters):
    names_count = (Counter(characters))
    names_to_detect = []

    for name_count in names_count:
        if names_count[name_count] > 2:
            names_to_detect.append(name_count)
    return names_to_detect

names_to_detect = remove_non_frequent_occurrences(characters)

def get_chars_to_detect(names_to_detect):
    chars_to_detect = []
    for name in names_to_detect:
        parsed_name = HumanName(name)
        if parsed_name.first != "" and parsed_name.first not in chars_to_detect:
            chars_to_detect.append(parsed_name.first)
        if parsed_name.last != ""  and parsed_name.last not in chars_to_detect:
            chars_to_detect.append(parsed_name.last)
        if parsed_name.first != "" and parsed_name.last != "":
            char_name = parsed_name.first + " " + parsed_name.last
            if char_name not in chars_to_detect:
                chars_to_detect.append(char_name)
        if parsed_name.title != "" and name not in chars_to_detect:
            chars_to_detect.append(name)
        if parsed_name.title != "" and parsed_name.first != "":
            char_name = parsed_name.title + " " + parsed_name.first
            if char_name not in chars_to_detect:
                chars_to_detect.append(char_name)
        if parsed_name.title != "" and parsed_name.last!= "":
            char_name = parsed_name.title + " " + parsed_name.last
            if char_name not in chars_to_detect:
                chars_to_detect.append(char_name)
        if parsed_name.title != "" and parsed_name.title not in chars_to_detect:
            chars_to_detect.append(parsed_name.title)

    chars_to_detect.sort()
    return chars_to_detect

final_chars_to_detect = (get_chars_to_detect(names_to_detect))


def remove_genitives_and_determiners_and_full_stop_titles(final_chars_to_detect):
    pattern = r"(\w+.)*\w+'s" or r"(\w+.)*\w+’s"
    final_chars_to_detect_always = []
    genitives = []
    matches = re.finditer(pattern, str(final_chars_to_detect))
    for match in matches:
        genitives.append(match.group())

    for name in final_chars_to_detect:
        if name not in genitives and name not in determiners and name not in 
titles_with_full_stop:
            final_chars_to_detect_always.append(name)
    return final_chars_to_detect_always

our_characters = remove_genitives_and_determiners_and_full_stop_titles(final_chars_to_detect)


###creating patterns
def create_trainig_data(chars_to_detect, type):
    data = chars_to_detect
    patterns=[]
    for item in data:
        pattern = {
                    "label": type,
                    "pattern": item
                    }
        patterns.append(pattern)
    return (patterns)

patterns = create_trainig_data(our_characters, "PERSON")


###adapted_pipeline
def generate_rules(patterns):
    nlp = English()
    ruler = nlp.add_pipe("entity_ruler")
    ruler.add_patterns(patterns)
    nlp.to_disk(model_name)

generate_rules(patterns)

nlp = spacy.load(model_name)
nlp.add_pipe('sentencizer')
doc = nlp(text)

aliases = []
for ent in doc.ents:
    aliases.append(str(ent))

aliases = list(set(aliases))


###identification of character occurrences’ interactions
def get_final_interaction_pairs():
    ents_per_sent = []
    for sent in doc.sents:
        if sent.ents != []:
            ent_per_sent = sent.ents
            ents_per_sent.append(ent_per_sent)

    interaction_pairs=[]
    for item in ents_per_sent:
        combine_pairs = list(combinations(item, 2))
        if combine_pairs != []:
            interaction_pairs.append(combine_pairs)

    final_interaction_pairs = []
    for item in interaction_pairs:
        for mention in item:
            final_interaction_pairs.append(mention)

    return final_interaction_pairs

final_interaction_pairs = (get_final_interaction_pairs())

###gender assignation
male_names = text_modul.convert_text_file_into_list(male_names_list)
female_names = text_modul.convert_text_file_into_list(female_names_list)
titles_M = text_modul.convert_text_file_into_list(male_titles_list)
titles_F= text_modul.convert_text_file_into_list(male_titles_lits)

male_titles = []
female_titles = []

for title in titles_M:
    if title in titles_and_honorifics:
        male_titles.append(title)

for title in titles_F:
    if title in titles_and_honorifics:
        female_titles.append(title)

def generate_gender(name, female_titles, male_titles, frequent_female_names,
frequent_male_names):
    parsed_name = HumanName(str(name))

    if parsed_name.title in female_titles:
        return "F"
    elif parsed_name.title in male_titles:
        return "M"
    else:
        if parsed_name.first in frequent_female_names and parsed_name.first in
frequent_male_names:
            return "U"
        elif parsed_name.first in frequent_female_names:
            return "F"
        elif parsed_name.first in frequent_male_names:
            return "M"
        else:
            return "U"

###matching algorithm - creating referents_list
character_names_unique = []
mergednames_and_gender = []


for alias in aliases:
    parsed_alias = HumanName(alias)
    gender = generate_gender(alias, female_titles, male_titles, female_names, male_names)

    char_name_and_gender = []
    char_name_first_and_gender = []
    char_name_last_and_gender = []
    if parsed_alias.title != "" and parsed_alias.first != "" and parsed_alias.last != "":
        char_name = alias
        char_name_and_gender.append(char_name)
        char_name_and_gender.append(gender)
        char_name_first_and_gender.append(parsed_alias.first)
        char_name_first_and_gender.append(gender)
        char_name_last_and_gender.append(parsed_alias.last)
        char_name_last_and_gender.append(gender)
        full_name = parsed_alias.first + " " + parsed_alias.last
        if full_name not in full_names:
            full_names.append(full_name)
        if char_name_and_gender != [] and char_name_and_gender not in character_names_unique:
            character_names_unique.append(char_name_and_gender)
        if char_name_first_and_gender not in mergednames_and_gender:
            mergednames_and_gender.append(char_name_first_and_gender)
        if char_name_last_and_gender not in mergednames_and_gender:
            mergednames_and_gender.append(char_name_last_and_gender)

for alias in aliases:
    parsed_alias = HumanName(alias)
    gender = generate_gender(alias, female_titles, male_titles, female_names, male_names)

    char_name_and_gender = []
    char_name_first_and_gender = []
    char_name_last_and_gender = []
    if parsed_alias.title == "" and parsed_alias.first != "" and parsed_alias.last != "" and
parsed_alias.first + " " + parsed_alias.last not in full_names:
        char_name = parsed_alias.first + " " + parsed_alias.last
        char_name_and_gender.append(char_name)
        char_name_and_gender.append(gender)
        char_name_first_and_gender.append(parsed_alias.first)
        char_name_first_and_gender.append(gender)
        char_name_last_and_gender.append(parsed_alias.last)
        char_name_last_and_gender.append(gender)
        if char_name not in full_names:
            full_names.append(char_name)
        if char_name_and_gender != [] and char_name_and_gender not in character_names_unique:
            character_names_unique.append(char_name_and_gender)
        if char_name_first_and_gender not in mergednames_and_gender:
            mergednames_and_gender.append(char_name_first_and_gender)
        if char_name_last_and_gender not in mergednames_and_gender:
            mergednames_and_gender.append(char_name_last_and_gender)

for alias in aliases:
    parsed_alias = HumanName(alias)
    gender = generate_gender(alias, female_titles, male_titles, female_names, male_names)

    char_name_and_gender = []
    char_name_first_and_gender = []

    if parsed_alias.title != "" and parsed_alias.first != "" and parsed_alias.last == "":
        char_name = parsed_alias.title + " " + parsed_alias.first
        char_name_and_gender.append(char_name)
        char_name_and_gender.append(gender)
        char_name_first_and_gender.append(parsed_alias.first)
        char_name_first_and_gender.append(gender)

        if char_name_first_and_gender not in mergednames_and_gender:
            if char_name_and_gender != [] and char_name_and_gender not in
character_names_unique:
                character_names_unique.append(char_name_and_gender)
                mergednames_and_gender.append(char_name_first_and_gender)

for alias in aliases:
    parsed_alias = HumanName(alias)
    gender = generate_gender(alias, female_titles, male_titles, female_names, male_names)

    char_name_and_gender = []
    char_name_last_and_gender = []

    if parsed_alias.title != "" and parsed_alias.first == "" and parsed_alias.last != "":
        char_name = parsed_alias.title + " " + parsed_alias.last
        char_name_and_gender.append(char_name)
        char_name_and_gender.append(gender)
        char_name_last_and_gender.append(parsed_alias.last)
        char_name_last_and_gender.append(gender)

        if char_name_last_and_gender not in mergednames_and_gender:
            if char_name_and_gender != [] and char_name_and_gender not in
character_names_unique:
                character_names_unique.append(char_name_and_gender)
                mergednames_and_gender.append(char_name_last_and_gender)

merged_names = []

for item in mergednames_and_gender:
    if item[0] not in merged_names:
        merged_names.append(item[0])

for alias in aliases:
    parsed_alias = HumanName(alias)
    gender = generate_gender(alias, female_titles, male_titles, female_names, male_names)

    char_name_and_gender = []
    char_name_first_or_last_and_gender = []

    if parsed_alias.title == "" and parsed_alias.first != "" and parsed_alias.last == "" and
parsed_alias.first not in merged_names:

        char_name = parsed_alias.first
        char_name_and_gender.append(char_name)
        char_name_and_gender.append(gender)
        char_name_first_or_last_and_gender.append(char_name)
        char_name_first_or_last_and_gender.append(gender)

        if char_name_and_gender != [] and char_name_and_gender not in character_names_unique:
            character_names_unique.append(char_name_and_gender)

###matching algorithm - unification of occurrences with their referents
names_count = Counter(aliases_multiple)

column_a = []
column_b = []

for item in final_interaction_pairs:
    parsed_item = HumanName(str(item[0]))
    item_gender = generate_gender(item[0], female_titles, male_titles, female_names,
male_names)

    variants = []
    for name in character_names_unique:
        parsed_character_name = HumanName(name[0])

        if parsed_item.title != "" and parsed_item.first != "" and parsed_item.last != "" and
parsed_item.first == parsed_character_name.first and parsed_item.last ==
parsed_character_name.last:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title == "" and parsed_item.first != "" and parsed_item.last != ""
and parsed_item.first == parsed_character_name.first and parsed_item.last ==
parsed_character_name.last:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title != "" and parsed_item.first == "" and parsed_item.last != ""
and parsed_item.last == parsed_character_name.last and item_gender == name[1]:
            if parsed_item.title == parsed_character_name.title:
                #print (str(name[0]) + " <--  " + str(item[0]))
                variants.append(name[0])

            if len(variants) == 0:
                if parsed_character_name.title == "":
                    #print (str(name[0]) + " <--  " + str(item[0]))
                    variants.append(name[0])

        elif parsed_item.title != "" and parsed_item.first == "" and parsed_item.last != ""
and parsed_item.last == parsed_character_name.first:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(name[0])

        elif parsed_item.title != "" and parsed_item.first != "" and parsed_item.last == ""
and parsed_item.first == parsed_character_name.first :#and parsed_item1.title ==
parsed_character_name.title and item1_gender == name[1]:
            print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title == "" and parsed_item.first != "" and parsed_item.last == ""
and parsed_item.first == parsed_character_name.first:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title == "" and parsed_item.first != "" and parsed_item.last == ""
and parsed_item.first == parsed_character_name.last:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(name[0])

        elif parsed_item.title != "" and parsed_item.first == "" and parsed_item.last == ""
and parsed_item.title == parsed_character_name.title:
            #print(str(name[0]) + " <--  " + str(item[0]))
            variants.append(name[0])

    if len(variants) > 1:
        #print(variants)
        count = 0
        variant_in_list=[]
        for variant in variants:
            frequency = names_count[variant]
            #print(frequency)
            if frequency > count:
                count = frequency
                most_frequented = variant
        #print(most_frequented)
        variant_in_list.append((most_frequented))
        column_a.append(variant_in_list)
    elif len(variants) == 1:
        column_a.append(variants)
    elif len(variants) == 0:
        org = []
        org.append(str(item[0]))
        column_b.append((org))


for item in final_interaction_pairs:
    parsed_item = HumanName(str(item[1]))
    item_gender = generate_gender(item[1], female_titles, male_titles, female_names,
male_names)

    variants = []
    for name in character_names_unique:
        parsed_character_name = HumanName(name[0])

        if parsed_item.title != "" and parsed_item.first != "" and parsed_item.last != "" and
parsed_item.first == parsed_character_name.first and parsed_item.last ==
parsed_character_name.last:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title == "" and parsed_item.first != "" and parsed_item.last != ""
and parsed_item.first == parsed_character_name.first and parsed_item.last ==
parsed_character_name.last:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title != "" and parsed_item.first == "" and parsed_item.last != ""
and parsed_item.last == parsed_character_name.last and item_gender == name[1]:

            if parsed_item.title == parsed_character_name.title:
                #print (str(name[0]) + " <--  " + str(item[0]))
                variants.append(name[0])

            if len(variants) == 0:
                if parsed_character_name.title == "":
                    #print (str(name[0]) + " <--  " + str(item[0]))
                    variants.append(name[0])


        elif parsed_item.title != "" and parsed_item.first == "" and parsed_item.last != ""
and parsed_item.last == parsed_character_name.first:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(name[0])
        elif parsed_item.title != "" and parsed_item.first != "" and parsed_item.last == ""
and parsed_item.first == parsed_character_name.first :#and parsed_item1.title ==
parsed_character_name.title and item1_gender == name[1]:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title == "" and parsed_item.first != "" and parsed_item.last == ""
and parsed_item.first == parsed_character_name.first:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(str(name[0]))

        elif parsed_item.title == "" and parsed_item.first != "" and parsed_item.last == ""
and parsed_item.first == parsed_character_name.last:
            #print (str(name[0]) + " <--  " + str(item[0]))
            variants.append(name[0])

        elif parsed_item.title != "" and parsed_item.first == "" and parsed_item.last == ""
and parsed_item.title == parsed_character_name.title:
            #print(str(name[0]) + " <--  " + str(item[0]))
            variants.append(name[0])

    if len(variants) > 1:
        #print(variants)
        count = 0
        variant_in_list=[]
        for variant in variants:
            frequency = names_count[variant]
            #print(frequency)
            if frequency > count:
                count = frequency
                most_frequented = variant
        #print(most_frequented)
        variant_in_list.append((most_frequented))
        column_b.append(variant_in_list)
    elif len(variants) == 1:
        column_b.append(variants)
    elif len(variants) == 0:
        org = []
        org.append(str(item[1]))
        column_b.append((org))

edges = zip(column_a, column_b)
edges = (tuple((edges)))

final_edges = []
for item in edges:
    if item[0] != item[1]:
        item_string = tuple((str(item[0]), str(item[1])))
        final_edges.append(item_string)

###graph extraction
interaction_pairs = final_edges

G = nx.Graph()
G.add_edges_from(interaction_pairs)

nx.write_gexf(G, graph_name)
```

Appendix C: Predefined list of titles and honorifics (*titles_and_honorifics_list*)

```
["10th", "1lt", "1sgt", "1st", "1stlt", "1stsgt", "2lt", "2nd", "2ndlt", "3rd", "4th", "5th",
"6th", "7th", "8th", "9th", "A1c", "Ab", "Abbess", "Abbot", "Abolitionist", "Academic",
"Acolyte", "Activist", "Actor ", "Actress", "Adept", "Adjutant", "Adm", "Admiral",
"Advertising", "Adviser", "Advocate", "Air", "Akhoond", "Alderman", "Almoner", "Ambassador",
"Amn", "Analytics", "Anarchist", "Animator", "Anthropologist", "Appellate", "Apprentice",
"Arbitrator", "Archbishop", "Archdeacon", "Archdruid", "Archduchess", "Archduke",
"Archeologist", "Architect", "Arhat", "Army", "Arranger", "Assistant", "Assoc", "Associate",
"Asst", "Astronomer", "Attache", "Attach\u00e9", "Attorney", "Aunt", "Auntie", "Author", "Award-winning",
"Ayatollah", "Baba", "Bailiff", "Ballet", "Bandleader", "Banker", "Banner", "Bard",
"Baron", "Baroness", "Barrister", "Baseball", "Bearer", "Behavioral", "Bench", "Bg", "Bgen",
"Biblical", "Bibliographer", "Biochemist", "Biographer", "Biologist", "Bishop", "Blessed",
"Blogger", "Blues", "Bodhisattva", "Bookseller", "Botanist", "Bp", "Brigadier", "Briggen",
"British", "Broadcaster", "Brother", "Buddha", "Burgess", "Burlesque", "Business",
"Businessman", "Businesswoman", "Bwana", "Canon", "Capt", "Captain", "Cardinal", "Cartographer",
"Cartoonist", "Catholicos", "Ccmsgt", "Cdr", "Celebrity", "Ceo", "Cfo", "Chair", "Chairs",
"Chancellor", "Chaplain", "Charg\u00e9 d'affaires", "Chef", "Cheikh", "Chemist", "Chief",
"Chieftain", "Choreographer", "Civil", "Classical", "Clergyman", "Clerk", "Cmsaf", "Cmsgt", "Co-chair",
"Co-chairs", "Co-founder", "Coach", "Col", "Collector", "Colonel", "Comedian",
"Comedienne", "Comic", "Commander", "Commander-in-chief", "Commodore", "Composer",
"Compositeur", "Comptroller", "Computer", "Comtesse", "Conductor", "Consultant", "Controller",
"Corporal", "Corporate", "Correspondent", "Councillor", "Counselor", "Count", "Countess",
"Courtier", "Cpl", "Cpo", "Cpt", "Credit", "Criminal", "Criminologist", "Critic", "Csm",
"Curator", "Customs", "Cwo-2", "Cwo-3", "Cwo-4", "Cwo-5", "Cwo2", "Cwo3", "Cwo4", "Cwo5",
"Cyclist", "Dame", "Dancer", "Dcn", "Deacon", "Delegate", "Deputy", "Designated", "Designer",
"Detective", "Developer", "Diplomat", "Dir", "Director", "Discovery", "Dissident", "District",
"Division", "Do", "Docent", "Docket", "Doctor", "Doyen", "Dpty", "Dr", "Dra", "Dramatist",
"Druid", "Drummer", "Duchesse", "Dutchess", "Ecologist", "Economist", "Editor", "Edmi",
"Edohen", "Educator", "Effendi", "Ekegbian", "Elerunwon", "Eminence", "Emperor", "Empress",
"Engineer", "English", "Ens", "Entertainer", "Entrepreneur", "Envoy", "Essayist", "Evangelist",
"Excellency", "Excellent", "Exec", "Executive", "Expert", "Fadm", "Family", "Father", "Federal",
"Field", "Film", "Financial", "First", "Flag", "Flying", "Foreign", "Forester", "Founder", "Fr",
"Friar", "Gaf", "Gen", "General", "Generalissimo", "Gentiluomo", "Giani", "Goodman", "Goodwife",
"Governor", "Graf", "Grand", "Group", "Guitarist", "Guru", "Gyani", "Gysgt", "Hajji", "Headman",
"Heir", "Heiress", "Her", "Hereditary", "High", "Highness", "His", "Historian", "Historicus",
"Historien", "Holiness", "Hon", "Honorable", "Honourable", "Host", "Illustrator", "Imam",
"Industrialist", "Information", "Instructor", "Intelligence", "Intendant", "Inventor",
"Investigator", "Investor", "Journalist", "Journeyman", "Jr", "Judge", "Judicial", "Junior",
"Jurist", "Keyboardist", "King", "King's", "Kingdom", "Knowledge", "Lady", "Lama", "Lamido",
"Law", "Lawyer", "Lcdr", "Lcpl", "Leader", "Lecturer", "Legal", "Librarian", "Lieutenant",
"Linguist", "Literary", "Lord", "Lt", "Ltc", "Ltcol", "Ltg", "Ltgen", "Ltjg", "Lyricist",
"Madam", "Madame", "Mademoiselle", "Mag", "Mag-judge", "Mag/judge", "Magistrate", "Magistrate-judge",
"Magnate", "Maharajah", "Maharani", "Mahdi", "Maid", "Maj", "Majesty", "Majgen",
"Manager", "Marcher", "Marchess", "Marchioness", "Marketing", "Marquess", "Marquis", "Marquise",
"Master", "Mathematician", "Mathematics", "Matriarch", "Mayor", "Mcpo", "Mcpoc", "Mcpon", "Md",
"Member", "Memoirist", "Merchant", "Met", "Metropolitan", "Mg", "Mgr", "Mgysgt", "Military",
"Minister", "Miss", "Misses", "Missionary", "Mister", "Mlle", "Mme", "Mobster", "Model", "Monk",
"Monsignor", "Most", "Mother", "Mountaineer", "Mpco-cg", "Mr", "Mrs", "Ms", "Msg", "Msgt",
"Mufti", "Mullah", "Municipal", "Murshid", "Musician", "Musicologist", "Mx", "Mystery", "Nanny",
"Narrator", "National", "Naturalist", "Navy", "Neuroscientist", "Novelist", "Nurse",
"Obstetritian", "Officer", "Opera", "Operating", "Ornithologist", "Painter", "Paleontologist",
"Pastor", "Patriarch", "Pediatrician", "Personality", "Petty", "Pfc", "Pharaoh", "Phd",
"Philantropist", "Philosopher", "Photographer", "Physician", "Physicist", "Pianist", "Pilot",
"Pioneer", "Pir", "Player", "Playwright", "Po1", "Po2", "Po3", "Poet", "Police", "Political",
"Politician", "Pope", "Prefect", "Prelate", "Premier", "Pres", "Presbyter", "President",
"Presiding", "Priest", "Priestess", "Primate", "Prime", "Prin", "Prince", "Princess",
"Principal", "Printer", "Printmaker", "Prior", "Private", "Pro", "Producer", "Prof",
"Professor", "Provost", "Pslc", "Psychiatrist", "Psychologist", "Publisher", "Pursuivant",
"Pv2", "Pvt", "Queen", "Queen's", "Rabbi", "Radio", "Radm", "Rangatira", "Ranger", "Rdml",
"Rear", "Rebbe", "Registrar", "Rep", "Representative", "Researcher", "Resident", "Rev",
"Revenue", "Reverend", "Right", "Risk", "Rock", "Royal", "Rt", "Sa", "Sailor", "Saint",
"Sainte", "Saoshyant", "Satirist", "Scholar", "Schoolmaster", "Scientist", "Scpo",
"Screenwriter", "Se", "Secretary", "Security", "Seigneur", "Senator", "Senior", "Senior-judge",
"Sergeant", "Servant", "Sfc", "Sgm", "Sgt", "Sgtmaj", "Sgtmajmc", "Shaik", "Shaikh", "Shayk",
"Shaykh", "Shehu", "Sheik", "Sheikh", "Shekh", "Sheriff", "Siddha", "Singer", "Singer-songwriter",
"Sir", "Sister", "Sma", "Smsgt", "Sn", "Soccer", "Social", "Sociologist",
"Software", "Soldier", "Solicitor", "Soprano", "Spc", "Speaker", "Special", "Sr", "Sra", "Srta",
"Ssg", "Ssgt", "St", "Staff", "State", "States", "Strategy", "Subaltern", "Subedar",
"Suffragist", "Sultan", "Sultana", "Superior", "Supreme", "Surgeon", "Swami", "Swordbearer",
"Sysselmann", "Tax", "Teacher", "Technical", "Technologist", "Television ", "Tenor", "Theater",
"Theatre", "Theologian", "Theorist", "Timi", "Tirthankar", "Translator", "Travel", "Treasurer",
"Tsar", "Tsarina", "Tsgt", "Uk", "Uncle", "United", "Us", "Vadm", "Vardapet", "Vc", "Venerable",
"Verderer", "Vicar", "Vice", "Viscount", "Vizier", "Vocalist", "Voice", "Warden", "Warrant",
"Wing", "Wm", "Wo-1", "Wo1", "Wo2", "Wo3", "Wo4", "Wo5", "Woodman", "Writer", "Zoologist",
"10th.", "1lt.", "1sgt.", "1st.", "1stlt.", "1stsgt.", "2lt.", "2nd.", "2ndlt.", "3rd.", "4th.",
"5th.", "6th.", "7th.", "8th.", "9th.", "A1c.", "Ab.", "Abbess.", "Abbot.", "Abolitionist.",
"Academic.", "Acolyte.", "Activist.", "Actor. ", "Actress.", "Adept.", "Adjutant.", "Adm.",
"Admiral.", "Advertising.", "Adviser.", "Advocate.", "Air.", "Akhoond.", "Alderman.",
"Almoner.", "Ambassador.", "Amn.", "Analytics.", "Anarchist.", "Animator.", "Anthropologist.",
"Appellate.", "Apprentice.", "Arbitrator.", "Archbishop.", "Archdeacon.", "Archdruid.",
"Archduchess.", "Archduke.", "Archeologist.", "Architect.", "Arhat.", "Army.", "Arranger.",
"Assistant.", "Assoc.", "Associate.", "Asst.", "Astronomer.", "Attache.", "Attach.\u00e9",
"Attorney.", "Aunt.", "Auntie.", "Author.", "Award.-winning.", "Ayatollah.", "Baba.",
"Bailiff.", "Ballet.", "Bandleader.", "Banker.", "Banner.", "Bard.", "Baron.", "Baroness.",
"Barrister.", "Baseball.", "Bearer.", "Behavioral.", "Bench.", "Bg.", "Bgen.", "Biblical.",
"Bibliographer.", "Biochemist.", "Biographer.", "Biologist.", "Bishop.", "Blessed.",
"Blogger.", "Blues.", "Bodhisattva.", "Bookseller.", "Botanist.", "Bp.", "Brigadier.",
"Briggen.", "British.", "Broadcaster.", "Brother.", "Buddha.", "Burgess.", "Burlesque.",
"Business.", "Businessman.", "Businesswoman.", "Bwana.", "Canon.", "Capt.", "Captain.",
"Cardinal.", "Cartographer.", "Cartoonist.", "Catholicos.", "Ccmsgt.", "Cdr.", "Celebrity.",
"Ceo.", "Cfo.", "Chair.", "Chairs.", "Chancellor.", "Chaplain.", "Charg.\u00e9 d.'affaires.",
"Chef.", "Cheikh.", "Chemist.", "Chief.", "Chieftain.", "Choreographer.", "Civil.",
"Classical.", "Clergyman.", "Clerk.", "Cmsaf.", "Cmsgt.", "Co.-chair.", "Co.-chairs.", "Co.-
founder.", "Coach.", "Col.", "Collector.", "Colonel.", "Comedian.", "Comedienne.", "Comic.",
"Commander.", "Commander.-in.-chief.", "Commodore.", "Composer.", "Compositeur.",
"Comptroller.", "Computer.", "Comtesse.", "Conductor.", "Consultant.", "Controller.",
"Corporal.", "Corporate.", "Correspondent.", "Councillor.", "Counselor.", "Count.",
"Countess.", "Courtier.", "Cpl.", "Cpo.", "Cpt.", "Credit.", "Criminal.", "Criminologist.",
"Critic.", "Csm.", "Curator.", "Customs.", "Cwo.-2.", "Cwo.-3.", "Cwo.-4.", "Cwo.-5.", "Cwo2.",
"Cwo3.", "Cwo4.", "Cwo5.", "Cyclist.", "Dame.", "Dancer.", "Dcn.", "Deacon.", "Delegate.",
"Deputy.", "Designated.", "Designer.", "Detective.", "Developer.", "Diplomat.", "Dir.",
"Director.", "Discovery.", "Dissident.", "District.", "Division.", "Do.", "Docent.", "Docket.",
"Doctor.", "Doyen.", "Dpty.", "Dr.", "Dra.", "Dramatist.", "Druid.", "Drummer.", "Duchesse.",
"Dutchess.", "Ecologist.", "Economist.", "Editor.", "Edmi.", "Edohen.", "Educator.", "Effendi.",
"Ekegbian.", "Elerunwon.", "Eminence.", "Emperor.", "Empress.", "Engineer.", "English.", "Ens.",
"Entertainer.", "Entrepreneur.", "Envoy.", "Essayist.", "Evangelist.", "Excellency.",
"Excellent.", "Exec.", "Executive.", "Expert.", "Fadm.", "Family.", "Father.", "Federal.",
"Field.", "Film.", "Financial.", "First.", "Flag.", "Flying.", "Foreign.", "Forester.",
"Founder.", "Fr.", "Friar.", "Gaf.", "Gen.", "General.", "Generalissimo.", "Gentiluomo.",
"Giani.", "Goodman.", "Goodwife.", "Governor.", "Graf.", "Grand.", "Group.", "Guitarist.",
"Guru.", "Gyani.", "Gysgt.", "Hajji.", "Headman.", "Heir.", "Heiress.", "Her.", "Hereditary.",
"High.", "Highness.", "His.", "Historian.", "Historicus.", "Historien.", "Holiness.", "Hon.",
"Honorable.", "Honourable.", "Host.", "Illustrator.", "Imam.", "Industrialist.",
"Information.", "Instructor.", "Intelligence.", "Intendant.", "Inventor.", "Investigator.",
"Investor.", "Journalist.", "Journeyman.", "Jr.", "Judge.", "Judicial.", "Junior.", "Jurist.",
"Keyboardist.", "King.", "King.'s.", "Kingdom.", "Knowledge.", "Lady.", "Lama.", "Lamido.",
"Law.", "Lawyer.", "Lcdr.", "Lcpl.", "Leader.", "Lecturer.", "Legal.", "Librarian.",
"Lieutenant.", "Linguist.", "Literary.", "Lord.", "Lt.", "Ltc.", "Ltcol.", "Ltg.", "Ltgen.",
"Ltjg.", "Lyricist.", "Madam.", "Madame.", "Mademoiselle.", "Mag.", "Mag.-judge.",
"Mag./judge.", "Magistrate.", "Magistrate.-judge.", "Magnate.", "Maharajah.", "Maharani.",
"Mahdi.", "Maid.", "Maj.", "Majesty.", "Majgen.", "Manager.", "Marcher.", "Marchess.",
"Marchioness.", "Marketing.", "Marquess.", "Marquis.", "Marquise.", "Master.",
"Mathematician.", "Mathematics.", "Matriarch.", "Mayor.", "Mcpo.", "Mcpoc.", "Mcpon.", "Md.",
"Member.", "Memoirist.", "Merchant.", "Met.", "Metropolitan.", "Mg.", "Mgr.", "Mgysgt.",
"Military.", "Minister.", "Miss.", "Misses.", "Missionary.", "Mister.", "Mlle.", "Mme.",
"Mobster.", "Model.", "Monk.", "Monsignor.", "Most.", "Mother.", "Mountaineer.", "Mpco.-cg.",
"Mr.", "Mrs.", "Ms.", "Msg.", "Msgt.", "Mufti.", "Mullah.", "Municipal.", "Murshid.",
"Musician.", "Musicologist.", "Mx.", "Mystery.", "Nanny.", "Narrator.", "National.",
"Naturalist.", "Navy.", "Neuroscientist.", "Novelist.", "Nurse.", "Obstetritian.", "Officer.",
"Opera.", "Operating.", "Ornithologist.", "Painter.", "Paleontologist.", "Pastor.",
"Patriarch.", "Pediatrician.", "Personality.", "Petty.", "Pfc.", "Pharaoh.", "Phd.",
"Philantropist.", "Philosopher.", "Photographer.", "Physician.", "Physicist.", "Pianist.",
"Pilot.", "Pioneer.", "Pir.", "Player.", "Playwright.", "Po1.", "Po2.", "Po3.", "Poet.",
"Police.", "Political.", "Politician.", "Pope.", "Prefect.", "Prelate.", "Premier.", "Pres.",
"Presbyter.", "President.", "Presiding.", "Priest.", "Priestess.", "Primate.", "Prime.",
"Prin.", "Prince.", "Princess.", "Principal.", "Printer.", "Printmaker.", "Prior.", "Private.",
"Pro.", "Producer.", "Prof.", "Professor.", "Provost.", "Pslc.", "Psychiatrist.",
"Psychologist.", "Publisher.", "Pursuivant.", "Pv2.", "Pvt.", "Queen.", "Queen.'s.", "Rabbi.",
"Radio.", "Radm.", "Rangatira.", "Ranger.", "Rdml.", "Rear.", "Rebbe.", "Registrar.", "Rep.",
"Representative.", "Researcher.", "Resident.", "Rev.", "Revenue.", "Reverend.", "Right.",
"Risk.", "Rock.", "Royal.", "Rt.", "Sa.", "Sailor.", "Saint.", "Sainte.", "Saoshyant.",
"Satirist.", "Scholar.", "Schoolmaster.", "Scientist.", "Scpo.", "Screenwriter.", "Se.",
"Secretary.", "Security.", "Seigneur.", "Senator.", "Senior.", "Senior.-judge.", "Sergeant.",
"Servant.", "Sfc.", "Sgm.", "Sgt.", "Sgtmaj.", "Sgtmajmc.", "Shaik.", "Shaikh.", "Shayk.",
"Shaykh.", "Shehu.", "Sheik.", "Sheikh.", "Shekh.", "Sheriff.", "Siddha.", "Singer.", "Singer.-
songwriter.", "Sir.", "Sister.", "Sma.", "Smsgt.", "Sn.", "Soccer.", "Social.", "Sociologist.",
"Software.", "Soldier.", "Solicitor.", "Soprano.", "Spc.", "Speaker.", "Special.", "Sr.",
"Sra.", "Srta.", "Ssg.", "Ssgt.", "St.", "Staff.", "State.", "States.", "Strategy.",
"Subaltern.", "Subedar.", "Suffragist.", "Sultan.", "Sultana.", "Superior.", "Supreme.",
"Surgeon.", "Swami.", "Swordbearer.", "Sysselmann.", "Tax.", "Teacher.", "Technical.",
"Technologist.", "Television. ", "Tenor.", "Theater.", "Theatre.", "Theologian.", "Theorist.",
"Timi.", "Tirthankar.", "Translator.", "Travel.", "Treasurer.", "Tsar.", "Tsarina.", "Tsgt.",
"Uk.", "Uncle.", "United.", "Us.", "Vadm.", "Vardapet.", "Vc.", "Venerable.", "Verderer.",
"Vicar.", "Vice.", "Viscount.", "Vizier.", "Vocalist.", "Voice.", "Warden.", "Warrant.",
"Wing.", "Wm.", "Wo.-1.", "Wo1.", "Wo2.", "Wo3.", "Wo4.", "Wo5.", "Woodman.", "Writer.",
"Zoologist."]
```

Appendix D: List of determiners

```
["a few", "few", "fewer", "fewest", "a little", "little", "another", "other", "a", "an", "all",
"any", "both", "each", "either", "enough", "every", "half", "her", "his", "its", "least", "less",
"many", "more", "most", "much", "my", "neither", "no", "our", "several", "some", "such", "that",
"the", "their", "these", "this", "those", "what", "which", "whose", "your"]
```

Appendix E: List of male and female titles and honorifics

List of male titles and honorifics (*male_titles_list*)

```
['Archduke', 'Baron', 'Count', 'Emperor', 'King', 'Lord', 'Maharajah', 'Marquess', 'Marquis',
'Master', 'Mister', 'Mr', 'Mr.', 'Prince', 'Sheikh', 'Sir', 'Sultan', 'Tsar', 'Viscount',
'Brother', 'Father', 'Uncle']
```

List of female titles and honorifics (*female_titles_list*)

```
['Archduchess', 'Baroness', 'Countess', 'Dame', 'Empress', 'Lady', 'Maharani', 'Marchioness',
'Marquise', 'Miss', 'Mrs', 'Mrs.', 'Ms', 'Ms.', 'Princess', 'Queen', 'Tsarina', 'Auntie',
'Aunt', 'Mother', 'Sister']
```

Appendix F: List of popular male and female names

List of popular male names (*male_names_list*)

```
['Liam', 'Noah', 'Oliver', 'Elijah', 'James', 'William', 'Benjamin', 'Lucas', 'Henry',
'Theodore', 'Jack', 'Levi', 'Alexander', 'Jackson', 'Mateo', 'Daniel', 'Michael', 'Mason',
'Sebastian', 'Ethan', 'Logan', 'Owen', 'Samuel', 'Jacob', 'Asher', 'Aiden', 'John', 'Joseph',
'Wyatt', 'David', 'Leo', 'Luke', 'Julian', 'Hudson', 'Grayson', 'Matthew', 'Ezra', 'Gabriel',
'Carter', 'Isaac', 'Jayden', 'Luca', 'Anthony', 'Dylan', 'Lincoln', 'Thomas', 'Maverick',
'Elias', 'Josiah', 'Charles', 'Caleb', 'Christopher', 'Ezekiel', 'Miles', 'Jaxon', 'Isaiah',
'Andrew', 'Joshua', 'Nathan', 'Nolan', 'Adrian', 'Cameron', 'Santiago', 'Eli', 'Aaron', 'Ryan',
'Angel', 'Cooper', 'Waylon', 'Easton', 'Kai', 'Christian', 'Landon', 'Colton', 'Roman', 'Axel',
'Brooks', 'Jonathan', 'Robert', 'Jameson', 'Ian', 'Everett', 'Greyson', 'Wesley', 'Jeremiah',
'Hunter', 'Leonardo', 'Jordan', 'Jose', 'Bennett', 'Silas', 'Nicholas', 'Parker', 'Beau',
'Weston', 'Austin', 'Connor', 'Carson', 'Dominic', 'Xavier', 'Jaxson', 'Jace', 'Emmett', 'Adam',
'Declan', 'Rowan', 'Micah', 'Kayden', 'Gael', 'River', 'Ryder', 'Kingston', 'Damian', 'Sawyer',
'Luka', 'Evan', 'Vincent', 'Legend', 'Myles', 'Harrison', 'August', 'Bryson', 'Amir',
'Giovanni', 'Chase', 'Diego', 'Milo', 'Jasper', 'Walker', 'Jason', 'Brayden', 'Cole',
'Nathaniel', 'George', 'Lorenzo', 'Zion', 'Luis', 'Archer', 'Enzo', 'Jonah', 'Thiago', 'Theo',
'Ayden', 'Zachary', 'Calvin', 'Braxton', 'Ashton', 'Rhett', 'Atlas', 'Jude', 'Bentley',
'Carlos', 'Ryker', 'Adriel', 'Arthur', 'Ace', 'Tyler', 'Jayce', 'Max', 'Elliot', 'Graham',
'Kaiden', 'Maxwell', 'Juan', 'Dean', 'Matteo', 'Malachi', 'Ivan', 'Elliott', 'Jesus',
'Emiliano', 'Messiah', 'Gavin', 'Maddox', 'Camden', 'Hayden', 'Leon', 'Antonio', 'Justin',
'Tucker', 'Brandon', 'Kevin', 'Judah', 'Finn', 'King', 'Brody', 'Xander', 'Nicolas', 'Charlie',
'Arlo', 'Emmanuel', 'Barrett', 'Felix', 'Alex', 'Miguel', 'Abel', 'Alan', 'Beckett', 'Amari',
'Karter', 'Timothy', 'Abraham', 'Jesse', 'Zayden', 'Blake', 'Alejandro', 'Dawson', 'Tristan',
'Victor', 'Avery', 'Joel', 'Grant', 'Eric', 'Patrick', 'Peter', 'Richard', 'Edward', 'Andres',
'Emilio', 'Colt', 'Knox', 'Beckham', 'Adonis', 'Kyrie', 'Matias', 'Oscar', 'Lukas', 'Marcus',
'Hayes', 'Caden', 'Remington', 'Griffin', 'Nash', 'Israel', 'Steven', 'Holden', 'Rafael',
'Zane', 'Jeremy', 'Kash', 'Preston', 'Kyler', 'Jax', 'Jett', 'Kaleb', 'Riley', 'Simon',
'Phoenix', 'Javier', 'Bryce', 'Louis', 'Mark', 'Cash', 'Lennox', 'Paxton', 'Malakai', 'Paul',
'Kenneth', 'Nico', 'Kaden', 'Lane', 'Kairo', 'Maximus', 'Omar', 'Finley', 'Atticus', 'Crew',
'Brantley', 'Colin', 'Dallas', 'Walter', 'Brady', 'Callum', 'Ronan', 'Hendrix', 'Jorge',
'Tobias', 'Clayton', 'Emerson', 'Damien', 'Zayn', 'Malcolm', 'Kayson', 'Bodhi', 'Bryan',
'Aidan', 'Cohen', 'Brian', 'Cayden', 'Andre', 'Niko', 'Maximiliano', 'Zander', 'Khalil', 'Rory',
'Francisco', 'Cruz', 'Kobe', 'Reid', 'Daxton', 'Derek', 'Martin', 'Jensen', 'Karson', 'Tate',
'Muhammad', 'Jaden', 'Joaquin', 'Josue', 'Gideon', 'Dante', 'Cody', 'Bradley', 'Orion',
'Spencer', 'Angelo', 'Erick', 'Jaylen', 'Julius', 'Manuel', 'Ellis', 'Colson', 'Cairo',
'Gunner', 'Wade', 'Chance', 'Odin', 'Anderson', 'Kane', 'Raymond', 'Cristian', 'Aziel',
'Prince', 'Ezequiel', 'Jake', 'Otto', 'Eduardo', 'Rylan', 'Ali', 'Cade', 'Stephen', 'Ari',
'Kameron', 'Dakota', 'Warren', 'Ricardo', 'Killian', 'Mario', 'Romeo', 'Cyrus', 'Ismael',
'Russell', 'Tyson', 'Edwin', 'Desmond', 'Nasir', 'Remy', 'Tanner', 'Fernando', 'Hector',
'Titus', 'Lawson', 'Sean', 'Kyle', 'Elian', 'Corbin', 'Bowen', 'Wilder', 'Armani', 'Royal',
'Stetson', 'Briggs', 'Sullivan', 'Leonel', 'Callan', 'Finnegan', 'Jay', 'Zayne', 'Marshall',
'Kade', 'Travis', 'Sterling', 'Raiden', 'Sergio', 'Tatum', 'Cesar', 'Zyaire', 'Milan', 'Devin',
'Gianni', 'Kamari', 'Royce', 'Malik', 'Jared', 'Franklin', 'Clark', 'Noel', 'Marco', 'Archie',
'Apollo', 'Pablo', 'Garrett', 'Oakley', 'Memphis', 'Quinn', 'Onyx', 'Alijah', 'Baylor', 'Edgar',
'Nehemiah', 'Winston', 'Major', 'Rhys', 'Forrest', 'Jaiden', 'Reed', 'Santino', 'Troy',
'Caiden', 'Harvey', 'Collin', 'Solomon', 'Donovan', 'Damon', 'Jeffrey', 'Kason', 'Sage',
'Grady', 'Kendrick', 'Leland', 'Luciano', 'Pedro', 'Hank', 'Hugo', 'Esteban', 'Johnny',
'Kashton', 'Ronin', 'Ford', 'Mathias', 'Porter', 'Erik', 'Johnathan', 'Frank', 'Tripp', 'Casey',
'Fabian', 'Leonidas', 'Baker', 'Matthias', 'Philip', 'Jayceon', 'Kian', 'Saint', 'Ibrahim',
'Jaxton', 'Augustus', 'Callen', 'Trevor', 'Ruben', 'Adan', 'Conor', 'Dax', 'Braylen', 'Kaison',
'Francis', 'Kyson', 'Andy', 'Lucca', 'Mack', 'Peyton', 'Alexis', 'Deacon', 'Kasen', 'Kamden',
'Frederick', 'Princeton', 'Braylon', 'Wells', 'Nikolai', 'Iker', 'Bo', 'Dominick', 'Moshe',
'Cassius', 'Gregory', 'Lewis', 'Kieran', 'Isaias', 'Seth', 'Marcos', 'Omari', 'Shane', 'Keegan',
'Jase', 'Asa', 'Sonny', 'Uriel', 'Pierce', 'Jasiah', 'Eden', 'Rocco', 'Banks', 'Cannon',
'Denver', 'Zaiden', 'Roberto', 'Shawn', 'Drew', 'Emanuel', 'Kolton', 'Ayaan', 'Ares', 'Conner',
'Jalen', 'Alonzo', 'Enrique', 'Dalton', 'Moses', 'Koda', 'Bodie', 'Jamison', 'Phillip', 'Zaire',
'Jonas', 'Kylo', 'Moises', 'Shepherd', 'Allen', 'Kenzo', 'Mohamed', 'Keanu', 'Dexter', 'Conrad',
'Bruce', 'Sylas', 'Soren', 'Raphael', 'Rowen', 'Gunnar', 'Sutton', 'Quentin', 'Jaziel',
'Emmitt', 'Makai', 'Koa', 'Maximilian', 'Brixton', 'Dariel', 'Zachariah', 'Roy', 'Armando',
'Corey', 'Saul', 'Izaiah', 'Danny', 'Davis', 'Ridge', 'Yusuf', 'Ariel', 'Valentino', 'Jayson',
'Ronald', 'Albert', 'Gerardo', 'Ryland', 'Dorian', 'Drake', 'Gage', 'Rodrigo', 'Hezekiah',
'Kylan', 'Boone', 'Ledger', 'Santana', 'Jamari', 'Jamir', 'Lawrence', 'Reece', 'Kaysen',
'Shiloh', 'Arjun', 'Marcelo', 'Abram', 'Benson', 'Huxley', 'Nikolas', 'Zain', 'Kohen', 'Samson',
'Miller', 'Donald', 'Finnley', 'Kannon', 'Lucian', 'Watson', 'Keith', 'Westin', 'Tadeo',
'Sincere', 'Boston', 'Axton', 'Amos', 'Chandler', 'Leandro', 'Raul', 'Scott', 'Reign',
'Alessandro', 'Camilo', 'Derrick', 'Morgan', 'Julio', 'Clay', 'Edison', 'Jaime', 'Augustine',
'Julien', 'Zeke', 'Marvin', 'Bellamy', 'Landen', 'Dustin', 'Jamie', 'Krew', 'Kyree', 'Colter',
'Johan', 'Houston', 'Layton', 'Quincy', 'Case', 'Atreus', 'Cayson', 'Aarav', 'Darius', 'Harlan',
'Justice', 'Abdiel', 'Layne', 'Raylan', 'Arturo', 'Taylor', 'Anakin', 'Ander', 'Hamza', 'Otis',
'Azariah', 'Leonard', 'Colby', 'Duke', 'Flynn', 'Trey', 'Gustavo', 'Fletcher', 'Issac', 'Sam',
'Trenton', 'Callahan', 'Chris', 'Mohammad', 'Rayan', 'Lionel', 'Bruno', 'Jaxxon', 'Zaid',
'Brycen', 'Roland', 'Dillon', 'Lennon', 'Ambrose', 'Rio', 'Mac', 'Ahmed', 'Samir', 'Yosef',
'Tru', 'Creed', 'Tony', 'Alden', 'Aden', 'Alec', 'Carmelo', 'Dario', 'Marcel', 'Roger', 'Ty',
'Ahmad', 'Emir', 'Landyn', 'Skyler', 'Mohammed', 'Dennis', 'Kareem', 'Nixon', 'Rex', 'Uriah',
'Lee', 'Louie', 'Rayden', 'Reese', 'Alberto', 'Cason', 'Quinton', 'Kingsley', 'Chaim',
'Alfredo', 'Mauricio', 'Caspian', 'Legacy', 'Ocean', 'Ozzy', 'Briar', 'Wilson', 'Forest',
'Grey', 'Joziah', 'Salem', 'Neil', 'Remi', 'Bridger', 'Harry', 'Jefferson', 'Lachlan', 'Nelson',
'Casen', 'Salvador', 'Magnus', 'Tommy', 'Marcellus', 'Maximo', 'Jerry', 'Clyde', 'Aron',
'Keaton', 'Eliam', 'Lian', 'Trace', 'Douglas', 'Junior', 'Titan', 'Cullen', 'Cillian', 'Musa',
'Mylo', 'Hugh', 'Tomas', 'Vincenzo', 'Westley', 'Langston', 'Byron', 'Kiaan', 'Loyal',
'Orlando', 'Kyro', 'Amias', 'Amiri', 'Jimmy', 'Vicente', 'Khari', 'Brendan', 'Rey', 'Ben',
'Emery', 'Zyair', 'Bjorn', 'Evander', 'Ramon', 'Alvin', 'Ricky', 'Jagger', 'Brock', 'Dakari',
'Eddie', 'Blaze', 'Gatlin', 'Alonso', 'Curtis', 'Kylian', 'Nathanael', 'Devon', 'Wayne',
'Zakai', 'Mathew', 'Rome', 'Riggs', 'Aryan', 'Avi', 'Hassan', 'Lochlan', 'Stanley', 'Dash',
'Kaiser', 'Benicio', 'Bryant', 'Talon', 'Rohan', 'Wesson', 'Joe', 'Noe', 'Melvin', 'Vihaan',
'Zayd', 'Darren', 'Enoch', 'Mitchell', 'Jedidiah', 'Brodie', 'Castiel', 'Ira', 'Lance',
'Guillermo', 'Thatcher', 'Ermias', 'Misael', 'Jakari', 'Emory', 'Mccoy', 'Rudy', 'Thaddeus',
'Valentin', 'Yehuda', 'Bode', 'Madden', 'Kase', 'Bear', 'Boden', 'Jiraiya', 'Maurice', 'Alvaro',
'Ameer', 'Demetrius', 'Eliseo', 'Kabir', 'Kellan', 'Allan', 'Azrael', 'Calum', 'Niklaus', 'Ray',
'Damari', 'Elio', 'Jon', 'Leighton', 'Axl', 'Dane', 'Eithan', 'Eugene', 'Kenji', 'Jakob',
'Colten', 'Eliel', 'Nova', 'Santos', 'Zahir', 'Idris', 'Ishaan', 'Kole', 'Korbin', 'Seven',
'Alaric', 'Kellen', 'Bronson', 'Franco', 'Wes', 'Larry', 'Mekhi', 'Jamal', 'Dilan', 'Elisha',
'Brennan', 'Kace', 'Van', 'Felipe', 'Fisher', 'Cal', 'Dior', 'Judson', 'Alfonso', 'Deandre',
'Rocky', 'Henrik', 'Reuben', 'Anders', 'Arian', 'Damir', 'Jacoby', 'Khalid', 'Kye', 'Mustafa',
'Jadiel', 'Stefan', 'Yousef', 'Aydin', 'Jericho', 'Robin', 'Wallace', 'Alistair', 'Davion',
'Alfred', 'Ernesto', 'Kyng', 'Everest', 'Gary', 'Leroy', 'Yahir', 'Braden', 'Kelvin',
'Kristian', 'Adler', 'Avyaan', 'Brayan', 'Jones', 'Truett', 'Aries', 'Joey', 'Randy', 'Jaxx',
'Jesiah', 'Jovanni', 'Azriel', 'Brecken', 'Harley', 'Zechariah', 'Gordon', 'Jakai', 'Carl',
'Graysen', 'Kylen', 'Ayan', 'Branson', 'Crosby', 'Dominik', 'Jabari', 'Jaxtyn', 'Kristopher',
'Ulises', 'Zyon', 'Fox', 'Howard', 'Salvatore', 'Turner', 'Vance', 'Harlem', 'Jair', 'Jakobe',
'Jeremias', 'Osiris', 'Azael', 'Bowie', 'Canaan', 'Elon', 'Granger', 'Karsyn', 'Zavier', 'Cain',
'Dangelo', 'Heath', 'Yisroel', 'Gian', 'Shepard', 'Harold', 'Kamdyn', 'Rene', 'Rodney',
'Yaakov', 'Adrien', 'Kartier', 'Cassian', 'Coleson', 'Ahmir', 'Darian', 'Genesis', 'Kalel',
'Agustin', 'Wylder', 'Yadiel', 'Ephraim', 'Kody', 'Neo', 'Ignacio', 'Osman', 'Aldo', 'Abdullah',
'Cory', 'Blaine', 'Dimitri', 'Khai', 'Landry', 'Palmer', 'Benedict', 'Leif', 'Koen', 'Maxton',
'Mordechai', 'Zev', 'Atharv', 'Bishop', 'Blaise', 'Davian']
```

List of popular female names (*female_names_list*)

```
['Olivia', 'Emma', 'Charlotte', 'Amelia', 'Ava', 'Sophia', 'Isabella', 'Mia', 'Evelyn',
'Harper', 'Luna', 'Camila', 'Gianna', 'Elizabeth', 'Eleanor', 'Ella', 'Abigail', 'Sofia',
'Avery', 'Scarlett', 'Emily', 'Aria', 'Penelope', 'Chloe', 'Layla', 'Mila', 'Nora', 'Hazel',
'Madison', 'Ellie', 'Lily', 'Nova', 'Isla', 'Grace', 'Violet', 'Aurora', 'Riley', 'Zoey',
'Willow', 'Emilia', 'Stella', 'Zoe', 'Victoria', 'Hannah', 'Addison', 'Leah', 'Lucy', 'Eliana',
'Ivy', 'Everly', 'Lillian', 'Paisley', 'Elena', 'Naomi', 'Maya', 'Natalie', 'Kinsley',
'Delilah', 'Claire', 'Audrey', 'Aaliyah', 'Ruby', 'Brooklyn', 'Alice', 'Aubrey', 'Autumn',
'Leilani', 'Savannah', 'Valentina', 'Kennedy', 'Madelyn', 'Josephine', 'Bella', 'Skylar',
'Genesis', 'Sophie', 'Hailey', 'Sadie', 'Natalia', 'Quinn', 'Caroline', 'Allison', 'Gabriella',
'Anna', 'Serenity', 'Nevaeh', 'Cora', 'Ariana', 'Emery', 'Lydia', 'Jade', 'Sarah', 'Eva',
'Adeline', 'Madeline', 'Piper', 'Rylee', 'Athena', 'Peyton', 'Everleigh', 'Vivian', 'Clara',
'Raelynn', 'Liliana', 'Samantha', 'Maria', 'Iris', 'Ayla', 'Eloise', 'Lyla', 'Eliza', 'Hadley',
'Melody', 'Julia', 'Parker', 'Rose', 'Isabelle', 'Brielle', 'Adalynn', 'Arya', 'Eden', 'Remi',
'Mackenzie', 'Maeve', 'Margaret', 'Reagan', 'Charlie', 'Alaia', 'Melanie', 'Josie', 'Elliana',
'Cecilia', 'Mary', 'Daisy', 'Alina', 'Lucia', 'Ximena', 'Juniper', 'Kaylee', 'Magnolia',
'Summer', 'Adalyn', 'Sloane', 'Amara', 'Arianna', 'Isabel', 'Reese', 'Emersyn', 'Sienna',
'Kehlani', 'River', 'Freya', 'Valerie', 'Blakely', 'Genevieve', 'Esther', 'Valeria',
'Katherine', 'Kylie', 'Norah', 'Amaya', 'Bailey', 'Ember', 'Ryleigh', 'Georgia', 'Catalina',
'Emerson', 'Alexandra', 'Faith', 'Jasmine', 'Ariella', 'Ashley', 'Andrea', 'Millie', 'June',
'Khloe', 'Callie', 'Juliette', 'Sage', 'Ada', 'Anastasia', 'Olive', 'Alani', 'Brianna',
'Rosalie', 'Molly', 'Brynlee', 'Amy', 'Ruth', 'Aubree', 'Gemma', 'Taylor', 'Oakley', 'Margot',
'Arabella', 'Sara', 'Journee', 'Harmony', 'Blake', 'Alaina', 'Aspen', 'Noelle', 'Selena',
'Oaklynn', 'Morgan', 'Londyn', 'Zuri', 'Aliyah', 'Jordyn', 'Juliana', 'Finley', 'Presley',
'Zara', 'Leila', 'Marley', 'Sawyer', 'Amira', 'Lilly', 'London', 'Kimberly', 'Elsie', 'Ariel',
'Lila', 'Alana', 'Diana', 'Kamila', 'Nyla', 'Vera', 'Hope', 'Annie', 'Kaia', 'Myla', 'Alyssa',
'Angela', 'Ana', 'Lennon', 'Evangeline', 'Harlow', 'Rachel', 'Gracie', 'Rowan', 'Laila',
'Elise', 'Sutton', 'Lilah', 'Adelyn', 'Phoebe', 'Octavia', 'Sydney', 'Mariana', 'Wren',
'Lainey', 'Vanessa', 'Teagan', 'Kayla', 'Malia', 'Elaina', 'Saylor', 'Brooke', 'Lola', 'Miriam',
'Alayna', 'Adelaide', 'Daniela', 'Jane', 'Payton', 'Journey', 'Lilith', 'Delaney', 'Dakota',
'Mya', 'Charlee', 'Alivia', 'Annabelle', 'Kailani', 'Lucille', 'Trinity', 'Gia', 'Tatum',
'Raegan', 'Camille', 'Kaylani', 'Kali', 'Stevie', 'Maggie', 'Haven', 'Tessa', 'Daphne',
'Adaline', 'Hayden', 'Joanna', 'Jocelyn', 'Lena', 'Evie', 'Juliet', 'Fiona', 'Cataleya',
'Angelina', 'Leia', 'Paige', 'Julianna', 'Milani', 'Talia', 'Rebecca', 'Kendall', 'Harley',
'Lia', 'Phoenix', 'Dahlia', 'Logan', 'Camilla', 'Thea', 'Jayla', 'Brooklynn', 'Blair',
'Vivienne', 'Hallie', 'Madilyn', 'Mckenna', 'Evelynn', 'Ophelia', 'Celeste', 'Alayah', 'Winter',
'Catherine', 'Collins', 'Nina', 'Briella', 'Palmer', 'Noa', 'Mckenzie', 'Kiara', 'Amari',
'Adriana', 'Gracelynn', 'Lauren', 'Cali', 'Kalani', 'Aniyah', 'Nicole', 'Alexis', 'Mariah',
'Gabriela', 'Wynter', 'Amina', 'Ariyah', 'Adelynn', 'Remington', 'Reign', 'Alaya', 'Dream',
'Alexandria', 'Willa', 'Avianna', 'Makayla', 'Gracelyn', 'Elle', 'Amiyah', 'Arielle', 'Elianna',
'Giselle', 'Brynn', 'Ainsley', 'Aitana', 'Charli', 'Demi', 'Makenna', 'Rosemary', 'Danna',
'Izabella', 'Lilliana', 'Melissa', 'Samara', 'Lana', 'Mabel', 'Everlee', 'Fatima', 'Leighton',
'Esme', 'Raelyn', 'Madeleine', 'Nayeli', 'Camryn', 'Kira', 'Annalise', 'Selah', 'Serena',
'Royalty', 'Rylie', 'Celine', 'Laura', 'Brinley', 'Frances', 'Michelle', 'Heidi', 'Rory',
'Sabrina', 'Destiny', 'Gwendolyn', 'Alessandra', 'Poppy', 'Amora', 'Nylah', 'Luciana', 'Maisie',
'Miracle', 'Joy', 'Liana', 'Raven', 'Shiloh', 'Allie', 'Daleyza', 'Kate', 'Lyric', 'Alicia',
'Lexi', 'Addilyn', 'Anaya', 'Malani', 'Paislee', 'Elisa', 'Kayleigh', 'Azalea', 'Francesca',
'Jordan', 'Regina', 'Viviana', 'Aylin', 'Skye', 'Daniella', 'Makenzie', 'Veronica', 'Legacy',
'Maia', 'Ariah', 'Alessia', 'Carmen', 'Astrid', 'Maren', 'Helen', 'Felicity', 'Alexa',
'Danielle', 'Lorelei', 'Paris', 'Adelina', 'Bianca', 'Gabrielle', 'Jazlyn', 'Scarlet',
'Bristol', 'Navy', 'Esmeralda', 'Colette', 'Stephanie', 'Jolene', 'Marlee', 'Sarai', 'Hattie',
'Nadia', 'Rosie', 'Kamryn', 'Kenzie', 'Alora', 'Holly', 'Matilda', 'Sylvia', 'Cameron',
'Armani', 'Emelia', 'Keira', 'Braelynn', 'Jacqueline', 'Alison', 'Amanda', 'Cassidy', 'Emory',
'Ari', 'Haisley', 'Jimena', 'Jessica', 'Elaine', 'Dorothy', 'Mira', 'Eve', 'Oaklee', 'Averie',
'Charleigh', 'Lyra', 'Madelynn', 'Angel', 'Edith', 'Jennifer', 'Raya', 'Ryan', 'Heaven', 'Kyla',
'Wrenley', 'Meadow', 'Carter', 'Kora', 'Saige', 'Kinley', 'Maci', 'Mae', 'Salem', 'Aisha',
'Adley', 'Carolina', 'Sierra', 'Alma', 'Helena', 'Bonnie', 'Mylah', 'Briar', 'Aurelia', 'Leona',
'Macie', 'Maddison', 'April', 'Aviana', 'Lorelai', 'Alondra', 'Kennedi', 'Monroe', 'Emely',
'Maliyah', 'Ailani', 'Madilynn', 'Renata', 'Katie', 'Zariah', 'Imani', 'Amber', 'Analia',
'Ariya', 'Anya', 'Emberly', 'Emmy', 'Mara', 'Maryam', 'Dior', 'Mckinley', 'Virginia', 'Amalia',
'Mallory', 'Opal', 'Shelby', 'Clementine', 'Remy', 'Xiomara', 'Elliott', 'Elora', 'Katalina',
'Antonella', 'Skyler', 'Hanna', 'Kaliyah', 'Alanna', 'Haley', 'Itzel', 'Cecelia', 'Jayleen',
'Kensley', 'Beatrice', 'Journi', 'Dylan', 'Ivory', 'Yaretzi', 'Meredith', 'Sasha', 'Gloria',
'Oaklyn', 'Sloan', 'Abby', 'Davina', 'Lylah', 'Erin', 'Reyna', 'Kaitlyn', 'Michaela', 'Nia',
'Fernanda', 'Jaliyah', 'Jenna', 'Sylvie', 'Miranda', 'Anne', 'Mina', 'Myra', 'Aleena', 'Alia',
'Frankie', 'Ellis', 'Kathryn', 'Nalani', 'Nola', 'Jemma', 'Lennox', 'Marie', 'Angelica',
'Cassandra', 'Calliope', 'Adrianna', 'Ivanna', 'Zelda', 'Faye', 'Karsyn', 'Oakleigh', 'Dayana',
'Amirah', 'Megan', 'Siena', 'Reina', 'Rhea', 'Julieta', 'Malaysia', 'Henley', 'Liberty',
'Leslie', 'Alejandra', 'Kelsey', 'Charley', 'Capri', 'Priscilla', 'Zariyah', 'Savanna',
'Emerie', 'Christina', 'Skyla', 'Macy', 'Mariam', 'Melina', 'Chelsea', 'Dallas', 'Laurel',
'Briana', 'Holland', 'Lilian', 'Amaia', 'Blaire', 'Margo', 'Louise', 'Rosalia', 'Aleah',
'Bethany', 'Flora', 'Kylee', 'Kendra', 'Sunny', 'Laney', 'Tiana', 'Chaya', 'Ellianna', 'Milan',
'Aliana', 'Estella', 'Julie', 'Yara', 'Rosa', 'Cheyenne', 'Emmie', 'Carly', 'Janelle', 'Kyra',
'Naya', 'Malaya', 'Sevyn', 'Lina', 'Mikayla', 'Jayda', 'Leyla', 'Eileen', 'Irene', 'Karina',
'Aileen', 'Aliza', 'Kataleya', 'Kori', 'Indie', 'Lara', 'Romina', 'Jada', 'Kimber', 'Amani',
'Liv', 'Treasure', 'Louisa', 'Marleigh', 'Winnie', 'Kassidy', 'Noah', 'Monica', 'Keilani',
'Zahra', 'Zaylee', 'Hadassah', 'Jamie', 'Allyson', 'Anahi', 'Maxine', 'Karla', 'Khaleesi',
'Johanna', 'Penny', 'Hayley', 'Marilyn', 'Della', 'Freyja', 'Jazmin', 'Kenna', 'Ashlyn',
'Florence', 'Ezra', 'Melany', 'Murphy', 'Sky', 'Marina', 'Noemi', 'Coraline', 'Selene',
'Bridget', 'Alaiya', 'Angie', 'Fallon', 'Thalia', 'Rayna', 'Martha', 'Halle', 'Estrella',
'Joelle', 'Kinslee', 'Roselyn', 'Theodora', 'Jolie', 'Dani', 'Elodie', 'Halo', 'Nala',
'Promise', 'Justice', 'Nellie', 'Novah', 'Estelle', 'Jenesis', 'Miley', 'Hadlee', 'Janiyah',
'Waverly', 'Braelyn', 'Pearl', 'Aila', 'Katelyn', 'Sariyah', 'Azariah', 'Bexley', 'Giana',
'Lea', 'Cadence', 'Mavis', 'Ila', 'Rivka', 'Jovie', 'Yareli', 'Bellamy', 'Kamiyah', 'Kara',
'Baylee', 'Jianna', 'Kai', 'Alena', 'Novalee', 'Elliot', 'Livia', 'Ashlynn', 'Denver',
'Emmalyn', 'Persephone', 'Marceline', 'Jazmine', 'Kiana', 'Mikaela', 'Aliya', 'Galilea',
'Harlee', 'Jaylah', 'Lillie', 'Mercy', 'Ensley', 'Bria', 'Kallie', 'Celia', 'Berkley', 'Ramona',
'Jaylani', 'Jessie', 'Aubrie', 'Madisyn', 'Paulina', 'Averi', 'Aya', 'Chana', 'Milana', 'Cleo',
'Iyla', 'Cynthia', 'Hana', 'Lacey', 'Andi', 'Giuliana', 'Milena', 'Leilany', 'Saoirse', 'Adele',
'Drew', 'Bailee', 'Hunter', 'Rayne', 'Anais', 'Kamari', 'Paula', 'Rosalee', 'Teresa', 'Zora',
'Avah', 'Belen', 'Greta', 'Layne', 'Scout', 'Zaniyah', 'Amelie', 'Dulce', 'Chanel', 'Clare',
'Rebekah', 'Giovanna', 'Ellison', 'Isabela', 'Kaydence', 'Rosalyn', 'Royal', 'Alianna',
'August', 'Nyra', 'Vienna', 'Amoura', 'Anika', 'Harmoni', 'Kelly', 'Linda', 'Aubriella',
'Kairi', 'Ryann', 'Avayah', 'Gwen', 'Whitley', 'Noor', 'Khalani', 'Marianna', 'Addyson',
'Annika', 'Karter', 'Vada', 'Tiffany', 'Artemis', 'Clover', 'Laylah', 'Paisleigh', 'Elyse',
'Kaisley', 'Veda', 'Zendaya', 'Simone', 'Alexia', 'Alisson', 'Angelique', 'Ocean', 'Elia',
'Lilianna', 'Maleah', 'Avalynn', 'Marisol', 'Goldie', 'Malayah', 'Emmeline', 'Paloma', 'Raina',
'Brynleigh', 'Chandler', 'Valery', 'Adalee', 'Tinsley', 'Violeta', 'Baylor', 'Lauryn',
'Marlowe', 'Birdie', 'Jaycee', 'Lexie', 'Loretta', 'Lilyana', 'Princess', 'Shay', 'Hadleigh',
'Natasha', 'Indigo', 'Zaria', 'Addisyn', 'Deborah', 'Leanna', 'Barbara', 'Kimora', 'Emerald',
'Raquel', 'Julissa', 'Robin', 'Austyn', 'Dalia', 'Nyomi', 'Ellen', 'Kynlee', 'Salma', 'Luella',
'Zayla', 'Addilynn', 'Giavanna', 'Samira', 'Amaris', 'Madalyn', 'Scarlette', 'Stormi', 'Etta',
'Ayleen', 'Brittany', 'Brylee', 'Araceli', 'Egypt', 'Iliana', 'Paityn', 'Zainab', 'Billie',
'Haylee', 'India', 'Kaiya', 'Nancy', 'Clarissa', 'Mazikeen', 'Taytum', 'Aubrielle', 'Rylan',
'Ainhoa', 'Aspyn', 'Elina', 'Elsa', 'Magdalena', 'Kailey', 'Arleth', 'Joyce', 'Judith',
'Crystal', 'Emberlynn', 'Landry', 'Paola', 'Braylee', 'Guinevere', 'Aarna', 'Aiyana', 'Kahlani',
'Lyanna', 'Sariah', 'Itzayana', 'Aniya', 'Frida', 'Jaylene', 'Kiera', 'Loyalty', 'Azaria',
'Jaylee', 'Kamilah', 'Keyla', 'Kyleigh', 'Micah', 'Nataly', 'Kathleen', 'Zoya', 'Meghan',
'Soraya', 'Zoie', 'Arlette', 'Zola', 'Luisa', 'Vida', 'Ryder', 'Tatiana', 'Tori', 'Aarya',
'Eleanora', 'Sandra', 'Soleil', 'Annabella']
```

Appendix G: Values of the eight network features for the shortened versions of novels

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	20	41	0.272	0.402	0.216	0.07	0.463	0.182
text_2	19	42	0.174	0.441	0.246	0.055	0.537	0.191
text_3	28	50	0.405	0.61	0.132	0.059	0.413	0.148
text_4	28	39	0.366	0.395	0.103	0.044	0.48	0.154
text_5	27	54	0.31	0.386	0.154	0.043	0.494	0.162
text_6	20	38	0.252	0.527	0.2	0.057	0.512	0.185
text_7	25	46	0.289	0.582	0.153	0.044	0.509	0.168
text_8	32	83	0.286	0.553	0.167	0.032	0.525	0.151
text_9	28	45	0.344	0.461	0.119	0.047	0.466	0.153
text_10	30	78	0.247	0.516	0.179	0.038	0.5	0.149
text_11	24	39	0.313	0.51	0.141	0.041	0.421	0.164
text_12	26	60	0.264	0.481	0.185	0.043	0.506	0.166
text_13	24	46	0.384	0.536	0.167	0.045	0.514	0.172
text_14	30	67	0.367	0.573	0.154	0.047	0.447	0.142
text_15	33	59	0.352	0.472	0.112	0.044	0.44	0.138

Selected network features extracted from the first 50,000 tokens of the 15 novels written by Author_1.

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	39	111	0.315	0.503	0.15	0.03	0.387	0.121
text_2	36	109	0.317	0.608	0.173	0.038	0.451	0.132
text_3	36	127	0.251	0.516	0.202	0.022	0.419	0.132
text_4	36	67	0.379	0.472	0.106	0.042	0.427	0.128
text_5	52	208	0.26	0.554	0.157	0.023	0.43	0.105
text_6	46	162	0.258	0.475	0.157	0.026	0.423	0.111
text_7	40	126	0.301	0.641	0.162	0.03	0.479	0.125
text_8	13	34	0.122	0.732	0.436	0.056	0.643	0.253
text_9	44	125	0.351	0.556	0.132	0.032	0.445	0.113
text_10	6	11	0	0.826	0.733	0.067	0.814	0.4
text_11	35	125	0.275	0.715	0.21	0.032	0.506	0.138
text_12	49	109	0.385	0.504	0.093	0.022	0.389	0.11
text_13	12	31	0.108	0.783	0.47	0.061	0.653	0.26
text_14	40	109	0.294	0.589	0.14	0.036	0.441	0.121
text_15	40	107	0.358	0.6	0.137	0.037	0.437	0.123

Selected network features extracted from the first 50,000 tokens of the 15 novels written by Author_2.

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	14	22	0.212	0.368	0.242	0.06	0.396	0.208
text_2	23	49	0.322	0.508	0.194	0.054	0.484	0.18
text_3	17	47	0.167	0.737	0.346	0.048	0.601	0.215
text_4	14	25	0.338	0.566	0.275	0.095	0.489	0.224
text_5	21	54	0.278	0.752	0.257	0.043	0.565	0.192
text_6	9	16	0.072	0.403	0.444	0.091	0.635	0.305
text_7	24	36	0.328	0.471	0.13	0.045	0.404	0.161
text_8	28	103	0.293	0.675	0.272	0.038	0.52	0.156
text_9	15	49	0.189	0.804	0.467	0.041	0.668	0.241
text_10	20	46	0.228	0.677	0.242	0.049	0.544	0.199
text_11	14	34	0.142	0.683	0.374	0.057	0.614	0.241
text_12	10	19	0.18	0.671	0.422	0.086	0.612	0.292
text_13	27	81	0.365	0.568	0.231	0.049	0.468	0.163
text_14	24	93	0.167	0.739	0.337	0.035	0.585	0.178
text_15	13	50	0.06	0.787	0.641	0.035	0.751	0.261

Selected network features extracted from the first 50,000 tokens of the 15 novels written by Author_3.

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	27	39	0.505	0.277	0.111	0.146	0.225	0.128
text_2	33	84	0.373	0.589	0.159	0.036	0.37	0.13
text_3	46	144	0.357	0.526	0.139	0.029	0.455	0.119
text_4	23	34	0.469	0.287	0.134	0.073	0.41	0.161
text_5	35	96	0.358	0.471	0.161	0.039	0.452	0.14
text_6	20	49	0.259	0.629	0.258	0.056	0.518	0.191
text_7	31	87	0.297	0.534	0.187	0.038	0.491	0.147
text_8	47	141	0.43	0.518	0.13	0.029	0.446	0.122
text_9	45	155	0.261	0.503	0.157	0.03	0.458	0.118
text_10	22	36	0.432	0.517	0.156	0.073	0.425	0.172
text_11	20	29	0.495	0.435	0.153	0.103	0.365	0.17
text_12	36	123	0.288	0.578	0.195	0.033	0.484	0.138
text_13	27	84	0.186	0.606	0.239	0.04	0.518	0.159
text_14	31	59	0.405	0.491	0.127	0.06	0.383	0.136
text_15	28	79	0.239	0.65	0.209	0.04	0.506	0.152

Selected network features extracted from the first 50,000 tokens of the 15 novels written by Author_4.

	Nodes	Edges	Modularity	Cluster. Coeff.	Degree	Betweenness	Closeness	Eigenvector
text_1	22	56	0.204	0.602	0.242	0.047	0.533	0.178
text_2	25	53	0.278	0.405	0.177	0.05	0.484	0.163
text_3	26	53	0.367	0.465	0.163	0.054	0.451	0.166
text_4	23	72	0.173	0.642	0.285	0.038	0.575	0.181
text_5	37	109	0.35	0.622	0.164	0.035	0.467	0.134
text_6	42	125	0.351	0.491	0.145	0.034	0.443	0.12
text_7	29	65	0.305	0.634	0.16	0.045	0.47	0.151
text_8	20	40	0.26	0.465	0.211	0.054	0.521	0.195
text_9	38	98	0.286	0.542	0.139	0.028	0.404	0.122
text_10	34	127	0.235	0.612	0.226	0.03	0.522	0.142
text_11	15	29	0.282	0.631	0.276	0.062	0.573	0.227
text_12	16	41	0.182	0.629	0.342	0.047	0.617	0.229
text_13	37	114	0.279	0.577	0.171	0.032	0.487	0.132
text_14	17	47	0.208	0.731	0.346	0.052	0.581	0.217
text_15	32	76	0.277	0.429	0.153	0.039	0.476	0.14

Selected network features extracted from the first 50,000 tokens of the 15 novels written by Author_5.

Appendix H: Source code for the extraction of the correlation matrix in R

```
library(PerformanceAnalytics)
d <- read.table(path, row.names=1, header=T)
round( cor(d), 2)

plot(d)

png(path, units = "in", width = 11, height=11, res = 600)
chart.Correlation(d,method = "spearman")
dev.off()
```

Appendix I: Values of the three selected network features for the shortened versions of the novels before standardization

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
text_1	0.402	0.07	0.463
text_2	0.441	0.055	0.537
text_3	0.61	0.059	0.413
text_4	0.395	0.044	0.48
text_5	0.386	0.043	0.494
text_6	0.527	0.057	0.512
text_7	0.582	0.044	0.509
text_8	0.553	0.032	0.525
text_9	0.461	0.047	0.466
text_10	0.516	0.038	0.5
text_11	0.51	0.041	0.421
text_12	0.481	0.043	0.506
text_13	0.536	0.045	0.514
text_14	0.573	0.047	0.447
text_15	0.472	0.044	0.44

Values of selected independent variables for Author_1.

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
text_1	0.503	0.03	0.387
text_2	0.608	0.038	0.451
text_3	0.516	0.022	0.419
text_4	0.472	0.042	0.427
text_5	0.554	0.023	0.43
text_6	0.475	0.026	0.423
text_7	0.641	0.03	0.479
text_8	0.732	0.056	0.643
text_9	0.556	0.032	0.445
text_10	0.826	0.067	0.814
text_11	0.715	0.032	0.506
text_12	0.504	0.022	0.389
text_13	0.783	0.061	0.653
text_14	0.589	0.036	0.441
text_15	0.6	0.037	0.437

Values of selected independent variables for Author_2.

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
text_1	0.368	0.06	0.396
text_2	0.508	0.054	0.484
text_3	0.737	0.048	0.601
text_4	0.566	0.095	0.489
text_5	0.752	0.043	0.565
text_6	0.403	0.091	0.635
text_7	0.471	0.045	0.404
text_8	0.675	0.038	0.52
text_9	0.804	0.041	0.668
text_10	0.677	0.049	0.544
text_11	0.683	0.057	0.614
text_12	0.671	0.086	0.612
text_13	0.568	0.049	0.468
text_14	0.739	0.035	0.585
text_15	0.787	0.035	0.751

Values of selected independent variables for Author_3.

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
text_1	0.277	0.146	0.225
text_2	0.589	0.036	0.37
text_3	0.526	0.029	0.455
text_4	0.287	0.073	0.41
text_5	0.471	0.039	0.452
text_6	0.629	0.056	0.518
text_7	0.534	0.038	0.491
text_8	0.518	0.029	0.446
text_9	0.503	0.03	0.458
text_10	0.517	0.073	0.425
text_11	0.435	0.103	0.365
text_12	0.578	0.033	0.484
text_13	0.606	0.04	0.518
text_14	0.491	0.06	0.383
text_15	0.65	0.04	0.506

Values of selected independent variables for Author_4.

	Avg. Clustering Coefficient	Avg. Betweenness Centrality	Avg. Closeness Centrality
text_1	0.602	0.047	0.533
text_2	0.405	0.05	0.484
text_3	0.465	0.054	0.451
text_4	0.642	0.038	0.575
text_5	0.622	0.035	0.467
text_6	0.491	0.034	0.443
text_7	0.634	0.045	0.47
text_8	0.465	0.054	0.521
text_9	0.542	0.028	0.404
text_10	0.612	0.03	0.522
text_11	0.631	0.062	0.573
text_12	0.629	0.047	0.617
text_13	0.577	0.032	0.487
text_14	0.731	0.052	0.581
text_15	0.429	0.039	0.476

Values of selected independent variables for Author_5.

Appendix J: Source code for the binary logistic regression analysis in R

```
library("psych")
library("ggplot2")

data <- read.table(path, sep="\t", row.names=1, header=T)

print(
  corr.test(data[,1:3], method = "spearman"),
  short = F)
# View of the features via PCA:
xy <- princomp(data[,1:3])$scores
xy <- as.data.frame(xy)
xy <- cbind(xy, y=as.factor(data$y))

ggplot(xy, aes(x=Comp.1,y=Comp.2, col=y)) +
  geom_point() +
  theme_bw()

data[,1:3] <- scale(data[,1:3])

fit     <- glm(y ~ ., data, family="binomial"
fitNull <- glm(y ~ 1, data, family="binomial")

anova(fit, fitNull, test = "Chisq")

testModel <- car::Anova(fit, type="III", test.statistic = "LR")
print(testModel)

interceptDummy <- c(0, 0, 0)
koeficienty    <- cbind( "Koef." = coefficients(fit) , confint(fit, level = 0.9833) )
expKoeficienty <- cbind( "Exp(Koef.)" = exp( coefficients(fit) ), exp( confint(fit, level =
0.9833) ) )
test           <- rbind( interceptDummy, as.matrix( testModel ))

output <- round( cbind( koeficienty, test, expKoeficienty)[-1,], 3)
print(koeficienty)

summary(fit)

prediction <- predict(fit, type="response")
classes <- prediction > 0.50

numberT <- sum( clases == data$y )
accuracy  <- round( numberT / nrow(data) * 100, 3 )
cat("Accuracy of prediction: ", accuracy, " %")
prediction
```