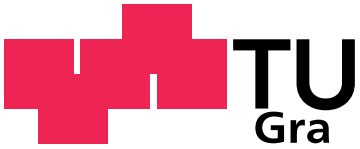

Klimashevskaia Anastasiia, Bsc

Automated Text Generation Driven By Data

Marshall Plan Scientific Report

submitted to

Austrian Marshall Plan Foundation

Supervisor
Assoc.Prof. Dipl.-Ing. Dr.techn. Christian Gütl

Institute of Interactive Systems and Data Science

Co-supervisor
Assoc.Prof. MSc Ph.D. Foaad Khosmood
California Polytechnic State University

Graz, August 2020

Abstract

Computer scientists have been attempting to tackle the task of text summarization for decades, introducing different techniques and solutions, broadening the experience in both extractive and abstractive summarization. However, the field of transcript text summarization appears to be less researched and fairly new. The methods of summarization for articles or other well-structured, grammatically correct texts are quite often not applicable in such a case at all or yield poor results. Moreover, transcripts with several speechmakers and various narratives require to take the speakers into consideration and keep track of the discourse. Numerous already existing solutions are very topic-specific and can be only applied to texts of a certain field. Lastly, a lot of the summaries produced with some of those techniques just appear to sound "robotic", especially the extractive summaries, where a coherent flow of sentences with smooth transitions between paragraphs is quite often missing.

This thesis suggests a novel approach to legislative proceedings transcript summarization using so-called "phenom" capturing technique in attempt to solve some of the aforementioned issues. A phenom is a specific pattern appearing in the text that is deemed to be worth of extracting and presenting in the summary. It can be a long back-and-forth discussion between two people, a pull-quote of interest, emotionally charged claim or a mention of a well-known person, organization or other entity. Those features tend to appear in certain parts of the text more often, thus a classification of text fragments has to be performed first to split the texts in certain chunks bearing different functions in the transcript. Luckily, legislative meetings are mainly quite consistent and well-structured in this sense with the organizers trying to stick to the agenda. After the parts of the text are classified and split into sections, the phenom extraction is performed, collecting facts to be filled into text templates crafted for each phenom. In the end, those generated sentences and paragraphs can be put together in the summary article, presented to the reader.

The whole system is build in a flexible way so the phenoms that the consumer is not interested in can be easily left or, if need be, other phenoms can be added and incorporated. Evaluation approaches, as well as further improvements and adaption for another application field are discussed in the chapters afterwards.

Acknowledgments

I want to first of all thank my parents for always believing in me and supporting me. Mom and dad, I would not have been where I am now without your help and love.

I am also thankful to both of my supervisors, Christian Gütl and Foaad Khosmood for making this thesis opportunity happen and involving me in such an interesting and exciting project. Both of them were very attentive and helpful no matter where I was - in Austria or in the US.

Thank you to my fellow Californian students who worked with me on this project - Thomas Gerrity, Richa Gadgil and Roberto Dominguez, who I actively collaborated with, shared the ideas and accomplished tasks together.

I would like to thank the Marshall Plan Scholarship Foundation ("Austrian Marshall Plan Foundation," 2020) for providing me with funds to make my trip to California possible. Without their financial support such a collaboration would not have been feasible for me and I am very glad that such an opportunity opened in front of me.

And last but not least, I want to thank my significant other - Stefan, thanks for motivating me for this adventure and always being by my side.

Contents

List of Figures

1 Introduction

In this chapter the motivation for this project will be provided. Furthermore, the contribution and the outline of this work will be defined afterwards.

1.1 Motivation

In 2015, the Institute for Advanced Technology and Public Policy (IATPP) launched the Digital Democracy project. The project was the first in history of the US to transcribe and make available the full legislative proceedings of the state of California's bicameral legislature. States of Texas, Florida and New York were subsequently covered as well. The system allows a full faceted search, and exploration of all the transcripts and search results can be viewed along with the corresponding video segments (See Figure 1.1).

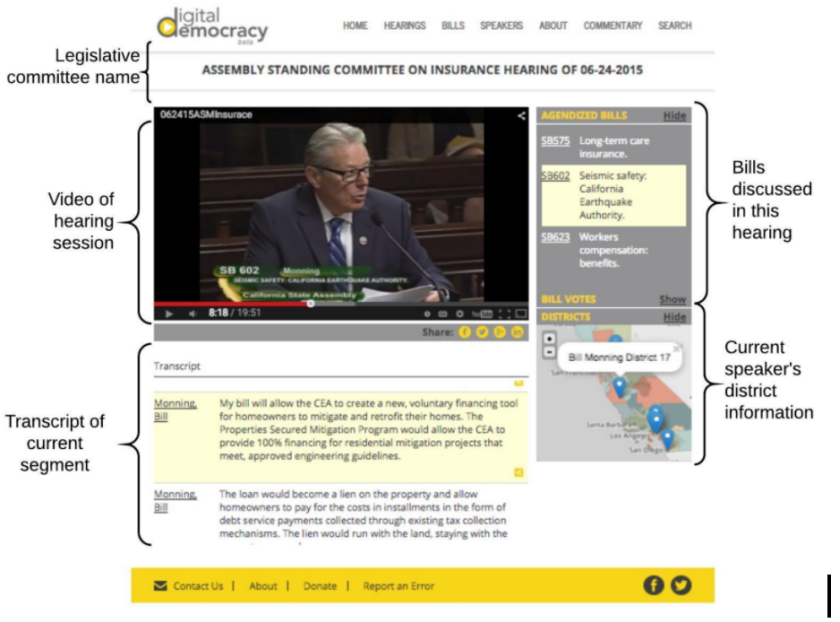

Figure 1.1: Digital Democracy project web interface.

The sheer volume and scale of these hearings makes it difficult for ordinary citizens to get a high level review of the events. The interactive search interface, however sophisticated, is not natural enough to convey a narrative. English-language summaries

are deemed more friendly and natural to most users. It is an interface they are very familiar with. This thesis now attempts to generate summaries that could end up on a news report for public consumption.

The IATPP is making available its unique dataset consisting of thousands of hours of human-verified video transcriptions, and associated metadata of legislative proceedings. California will be used for the main experimentations with other states later included in the system after it is proved to provide positive results.

1.2 Contribution

The main contribution of this work is the development of a summarization tool utilizing the Digital Democracy Database resources, providing more clarity and accessibility to all the assets regarding legislation proceeding in California. This work is an attempt to combine extractive and abstractive approaches to text summarization, which fused into emergence of the phenom extraction technique. Such a methodology allows easy expansion and is adaptable to changes required by the end user, which can lead to the creation of personalized legislation news generation resource.

Some solutions for various important subtasks are offered in this thesis as well, such as paragraph classification, article planning system and name repetition resolution. All these tasks allow the resulting summaries to reach closer to the quality and fluency of human-written abstracts - the gold standard that so many computer scientists and linguists have been striving to achieve.

Finally, a procedure to test the quality of the summaries in a user study is suggested in this thesis as well, due to it being impossible to estimate the accuracy and efficiency of the tool through commonly used metrics like ROUGE and others. A user study workflow is proposed and described in the end of this thesis to give an estimation opportunity for the results of the work.

1.3 Structure Of The Work

The structure of this work goes as following: the background and related works are discussed in the Chapter 2, giving information and overviews on such topics as news media and computational journalism, natural language processing, text summarization and spoken language summarization in particular with a discussion on the recent discoveries in the last topic. Chapter 3 gives an introduction to the concepts used in this work, as well as lists the requirements both functional and non-functional, finishing with listing the proposed tools and frameworks that were utilized. Chapter 4 expands more first on the architecture of the project, describing the components and the workflow, following with technical details on the implementation of the said components.

Chapter 6 draws conclusions on the accomplished tasks and outlines prospects of future work possible within the project bringing additions and improvements to the system.

2 Background and Related Work

In this chapter, the literature findings on the topic of the thesis are presented and various approaches on text summarization are discussed. Firstly, a general discussion on online news sources and especially legislative news representation is being held, contemplating on the current state of art, pros and cons of human-made articles and machine-produced automated summaries. A short introduction in news writing theory with information about common article structure is described with a following conversation on the topic of computational journalism, it's impacts on the news production and prospects. Secondly, the Natural Language Processing (NLP) definition and its tasks are given, bringing a short summary on the history of this field. One of the NLP tasks - text summarization is looked into, providing an overview on summary classification and methods for both singe- and multi-document text summarization. Lastly, the dialogue summarization problem is presented with a discussion and a review of the techniques applied for solving this problem. Most of the statistics brought up in the following subchapters will be U.S. based due to the relevance of the project to the United States legislative system representation.

2.1 News Media

> In the age of technology there is constant access to vast amounts of information. The basket overflows; people get overwhelmed; the eye of the storm is not so much what goes on in the world, it is the confusion of how to think, feel, digest, and react to what goes on.
>
> Criss Jami, *"Venus In Arms"*

The sources for getting news nowadays offer a great variety: TV broadcasting, newspapers, online forums, email subscriptions, social networks and so on. The leading sources among them remain to be television and the Internet - almost as many people now prefer to be informed online as those, who still like to get their news on TV, which is roughly four-in-ten among the U.S. citizen as can be seen in Fig.2.1. Also according to PEW Research Center (2009), the rise of the online news consumption is only growing. Bigger consumption could mean greater production - however, according to the statistics in the U.S. (see Fig.2.2) the number of journalists hired by news outlets such as newspapers and radio stations is plummeting dramatically, and there is a general declining trend in the number of employees in the field (Grieco, 2020). Thus, there

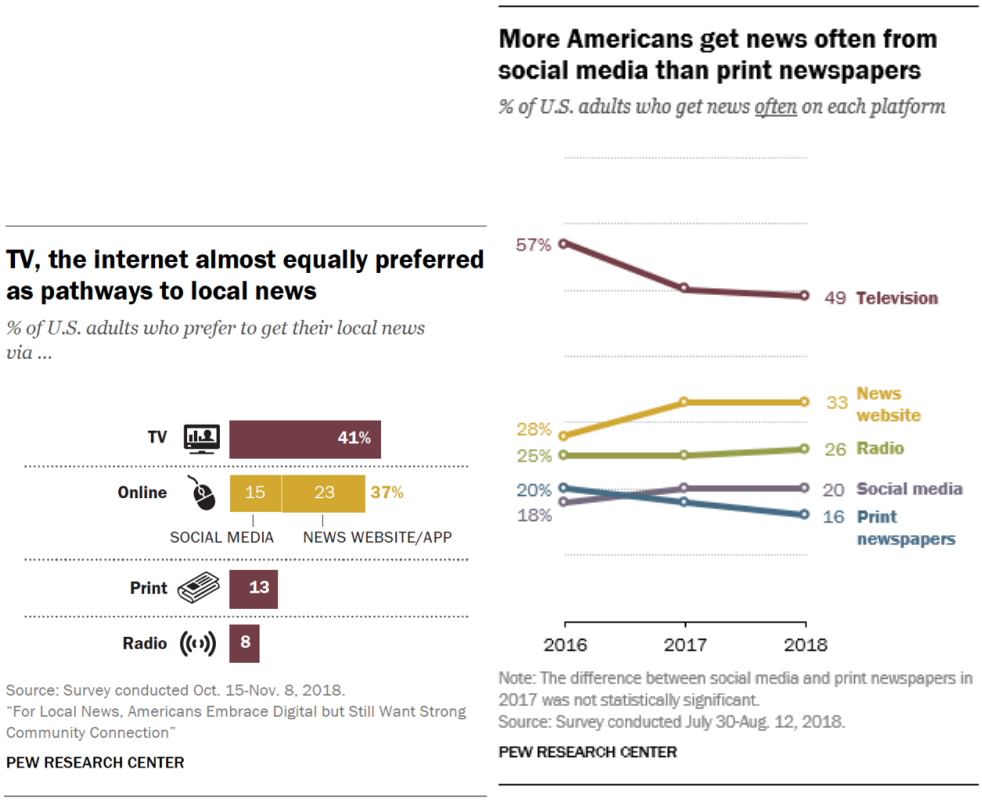

Figure 2.1: Infographics for news consumption in the U.S. (PEW Research Center, 2009)

is even less people to supply the increasing news production with more articles and stories.

There is another downside to the constantly growing news flow - the amount of information everywhere around is becoming overwhelming. *"NYTimes.com publishes roughly 150 articles a day (Monday-Saturday), 250 articles on Sunday and 65 blog posts per day"* (Meyer, 2016) - and this is only one newspaper in one country. Many people feel like they are struggling to keep up to date with all the things constantly happening and feel overwhelmed by the amount of news (Gottfried, 2020), which can even later discourage them from trying to stay involved at all.

2.1.1 The Structure Of The News Article

To understand how the news works and to build own news summary article it is crucial to know how a news article is organized, to keep in mind a classic structure of it and think about how an ordinary human-journalist would have written one. Even though each news reporter may have their own writing style, the basics remain almost always the same for most of the news reports.

Newsroom employees by news industry, 2008 to 2019

Number of U.S. newsroom employees in each news industry

Year	Total	Newspaper publishers	Broadcast television	Digital-native	Radio broadcasting	Cable television
2008	114,260	71,070	28,390	7,400	4,570	2,830
2009	104,490	60,770	28,040	8,090	4,330	3,260
2010	98,680	55,260	28,640	8,090	4,100	2,590
2011	97,350	54,050	28,050	9,520	3,540	2,190
2012	95,770	51,430	27,830	10,750	3,610	2,150
2013	92,240	48,920	25,650	11,250	3,700	2,720
2014	89,820	46,310	26,300	11,180	3,820	2,210
2015	90,400	44,120	28,430	11,710	3,380	2,760
2016	89,220	42,450	28,190	12,830	3,190	2,560
2017	87,630	39,210	28,900	13,260	3,320	2,940
2018	86,100	37,900	28,670	13,470	3,370	2,690
2019	87,510	34,950	30,120	16,090	3,530	2,820

Note: The OES survey is designed to produce estimates by combining data collected over a three-year period. Newsroom employees include news analysts, reporters and journalists; editors; photographers; and television, video and film camera operators and editors. Digital-native sector data is based on "other information services" industry code, whose largest segment is "internet publishing and broadcasting and web search portals."

Source: Pew Research Center analysis of Bureau of Labor Statistics Occupational Employment Statistics data.

PEW RESEARCH CENTER

Figure 2.2: Newsroom employees by news industries, 2008 to 2019 (Grieco, 2020)

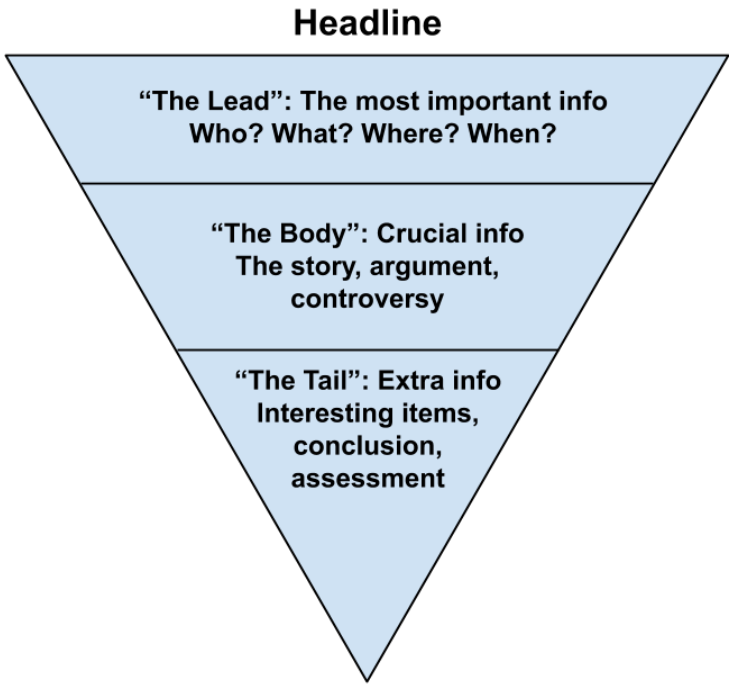

Figure 2.3: The "Inverted Pyramid", a news writing approach

When it comes to reading news online, many readers don't even get to the middle of the article (Manjoo, 2013). To help grab the attention of the reader, the journalist has to follow the main ABC of news writing - Accuracy, Brevity and Clarity (Parks, 2014). An approach to captivate the audience from the very beginning is to apply one of the most popular writing techniques, the so-called "inverted pyramid" (Pöttker, 2003). As it can be seen in Fig.2.3, the most important information contained in the article goes in the first paragraph, telling the reader all the essential facts: "What happened? Who did what? When was it?". Effectively, a brief summary can be narrowed down to only this part of the article. The second paragraph expands more with additional facts about controversies, discussion, further information, quotes, etc. In the end everything is rounded up with a paragraph with conclusions, assessments, some links to sources or further information.

A responsible journalist has to supply the facts they give in the article with the sources - whether it is a formal report from experts, police or other officials, or a person that was interviewed on the event, or a Web page. There are several ways to attach the source: it can be inserted directly afterwards in the brackets, or some web-pages adopt an approach with pop-up lines appearing if the user hovers over the sentence, or the sources can be all listed at the end of the article. This improves the integrity of the article greatly, convincing the reader to trust the journalist in their storytelling. There has been a long ongoing debate about which sources exactly can be called credible and can be used for news reporting (Franklin & Carlson, 2010), but the situation is constantly changing with the rise of Internet and since more and more people turn to it as the ultimate news source.

An article supplied with images like photos, plots, infographics also help to capture the attention of the reader. Furthermore, it helps even to engage the readers with less or no prior knowledge of the topic discussed in the article (Lee & Kim, 2016). Visualizations assist in understanding of more complicated trends and numbers that might be brought up by the journalist. Studies (Henke, Leissner, & Möhring, 2020) also have shown, that the presence of visual material in an article improves it's credibility from the reader's point of view as well.

2.1.2 Legislative News

> Much more of the answer, though, involves democracy itself. How can citizens govern themselves if they are unable to hold their governments accountable?
>
> Cohen, Hamilton, and Turner, *"Computational journalism"*

Legislative news might contain a lot of field-specific terms, which makes the news piece harder to consume. However, it is essential to keep the general public updated

and informed about what is happening in the government, what laws and bills are being accepted or rejected. The main principle of democracy is involving people in ruling the country - *"the concept of government legitimacy implies that citizens have some knowledge of their representative institution and a certain level of support for it."* (Kurtz, 1997).

The general aim of government transparency is to provide the means to the citizen to keep track of the decisions of the officials and be able to hold them accountable (Dawes & Helbig, 2010), which on its own can be a challenging quest (Blakeslee et al., 2015). Current developments in government transparency brought up various terms such as civic tech (Boehner & DiSalvo, 2016), E-Democracy (Parliamentary Office of Science and Technology, 2009), Open Government or Open Government Movement (Lathrop & Ruma, 2010; Latner, Dekhtyar, Khosmood, Angelini, & Voorhees, 2017). In the meantime, more and more initiatives are emerging to supply the people with insights of what is happening in the legal and political spheres, many countries or states provide portals accessible to anybody with information about hearings, meetings, changes and adoptions of laws ("California Legislative Information," 2020; "Eur-LEX: Access To European Union Law," 2020; "UK Legislation Portal," 2020). Those portals contain all the official information about legislation and accompanying documents with remarks and explanations.

However, a lot of interesting and important information can be missed due to the hearings and meetings mainly not being fully documented. Many news agencies don't have enough resources to send their reporters to assembly meetings to cover the happenings there (Matsa & Boyles, 2014). Meaning, the only way to know about the happenings is either only use factual information provided by the government, such as voting results, law details and information about the legislators, or have a journalist to look through the available recordings of the hearings to get more special insights. Yet writing an article based on such materials requires time, effort and knowledge of the domain. A journalist might have to look through hours of recordings to try and spot something particular in the video just to add up several sentences to the article in the end. Besides, all the information that the journalist have to browse through to connect all dots might not be even presented in one place - this adds up even more to the working time and cost of such work.

A big topic for discussion in political news in general is bias. It is no secret that some of the politicians always try to use the media and news for their own agenda, hoping to be represented in a better light (Karen Callaghan, 2001). In the past several decades the accusations of the news media being biased when it comes to politics have intensified immensely (Niven, 2002). While the politicians are blaming the journalist with prejudice and subjectivity, and the journalists are accusing each other, the public is becoming more skeptical about the news regarding laws, politics and legislature (Crawford, 2006). As soon as the citizen get disengaged from the politics and the lawmaking, the whole principle of the open government - *"establish a system of transparency, public participation,*

and collaboration" (McDermott, 2010) - is under risk. That is why it is essential for the authorities not only to provide all the information about actions and decisions that they make, but also try to get the ordinary civilian involved, make them want to take part in building their democracy and helping organizing their own country.

2.1.3 Computational Journalism

> Sometimes the question is asked: Is there an algorithm for journalism? The answer is yes, but to a certain degree.
>
> Linden et al., *"Algorithms for journalism: The future of news work"*

The term "computational journalism" was defined by Turner and Hamilton (2009) as *"the combination of algorithms, data, and knowledge from the social sciences to supplement the accountability function of journalism"*. Various other terms that technically refer to the same concept were also coined in the meantime: "algorithmic news", "automated content", "robot journalism" (Anderson, 2013; Dawson, 2010; Levy, 2018; Van Dalen, 2012) and so on. Essentially, they all mean either assistance of an intelligent system to reporters in writing news article - whether in data collection, drafting, analysis or content selection - or completely replacing a human reporter in generating simple reports filled with information from big databases which can be done instantly upon receiving new data.

Using Artificial Intelligence (AI) in reporting and automated content is not a complete novelty. It has been first used for some easier tasks like weather forecasting (Goldberg, Driedger, & Kittredge, 1994), financial and business reports (Yu, 2014), stock news (Nesterenko, 2016) and sports coverage (Schonfield, 2010). The data supplying this type of news is mainly bias-free, strictly organized and numeric, which allowed simple template-based generation approach. With computers gaining more power and being capable of doing unimaginably complicated calculations, the restrictions are being constantly lifted and more complex storytelling systems using neural networks and machine learning emerge now and then in various fields of application.

Arguably the biggest question in this field is the one of ethics. One might think that completely replacing human reporters might put the latter at risk or at least make them worried. Journalists indeed have proven to be a community protective of the boundaries of their profession (Lahav & Reich, 2011). In general, it is a big psychological factor to any person to perceive the likes of themselves as "us", while the rest alien, coming from the outside remains "them" (Brewer, 1999). In case of human journalism "they" are the AI reporters, so it is absolutely natural for the journalists to have more trust and a positive attitude towards their own community, while the "outgroup" of algorithms would receive more of a negative attitude. Moreover, some

fears are expressed that even if AI is going to be engaged only for some easy subroutine tasks, that might prevent young specialists from entering the job easier (Linden et al., 2017) - ordinarily it would be the beginner's work replaced then by the algorithm.

As for the ordinary reader's perception of a robot-written article, the same tendency remains, as it was shown by Graefe, Haim, Haarmann, and Brosius (2018). The reader mainly prefers human-written texts over machine-created, but interestingly in the case of some actually algorithm-generated articles being intentionally labeled as "human-authored" the text would still get the approval of the reader. Thus, possibly the readers still have less confidence with automatically generated articles and scrutinize them more than they would normally do reading an ordinary newspaper. Another interesting point proven by this study was the fact, that the readers brand machine-produced text as more credible, perchance due to the heavy use of numbers and precise facts in such articles, which gives the impression of reliability to the reader. The text created by the algorithm will have less calculation errors or misspellings than the human-made article (Linden et al., 2017).

On the other hand, not the whole journalistic community is being hostile to the innovations. Van Dalen (2012) has studied articles and Internet blog posts mentioning an automated sport news generating portal Statsheet and discovered, that the human community is not fully rejecting such a novelty. Main reasons for keeping self-confidence for the reporters, according to Van Dalen (2012) were such points as the AI journalism being still fairly abstract and not directly affecting them, or the fact that AI mainly occupies the fields of not the highest interest to the journalists.

Indeed, the emergence of automated news generation systems can be perceived not as a threat to the reporters' jobs, but as an opportunity for them *"to spend more time on substantive work"* (Peiser, 2019). Among the benefits of automated journalism is not only the dramatic decrease of human and time resources, but also a possible improvement in credibility, trust from the reader. It has been suggested already that computational approach to journalism can help with such issues as gender (Fischer-Hwang, Grosz, Hu, Karthik, & Yang, 2020) or political bias (Leppänen, Tuulonen, Sirén-Heikel, et al., 2020). As already mentioned above, the reader actually deems a machine-written text to be more credible and trustworthy, which could be a great advantage is such controversial topics as politics, elections or debates.

In general, the idea of an AI fully replacing journalists is still highly debatable, and various computer scientists and journalists are still very skeptical about it - like Linden et al. (2017) mentioned *"However, the idea that machines will become smart enough to replace journalists is [...] out of the question. [...] Algorithms only work on structured data. That's it. They only work on structured inputs. That's true of any computer. You can't take unstructured inputs and structure them on the fly."*

2.2 Natural Language Processing Problem

> NLP research has evolved from the era of punch cards and batch processing, in which the analysis of a sentence could take up to 7 minutes, to the era of Google and the likes of it, in which millions of webpages can be processed in less than a second.
>
> Young, Hazarika, Poria, and Cambria, *"Recent trends in deep learning based natural language processing"*

Natural Language Processing or computational linguistics is an aspect of Artificial Intelligence helping to establish communication between computers and humans (Reshamwala, Mishra, & Pawar, 2013), to understand human language, form sentences in it to communicate with users and provide information requested in a more natural way for the people. Additionally, NLP can be used also as an aid in human-human interaction (Hirschberg & Manning, 2015) - in the field of machine translation is one of those applications. The language perceived by the algorithm can be both in written and spoken form, processed and interpreted by the AI system. Over the last 20 years different NLP tasks have attracted the interest of many scientists, from programmers to linguists, statisticians and mathematicians. The applications of NLP can be met in various study fields, assisting doctors, scholars, people with disabilities, ordinary computer users. As Bird, Klein, and Loper (2009) remarks, *"NLP is important for scientific, economic, social, and cultural reasons"*.

The very first approaches in NLP were mainly using some hard-coded rules (Hayes-Roth, 1985). Such an approach was generally adopted after Chomsky (1957) proposed the concept of the rule-based descriptions of syntactic structures. This idea was instantly accepted in the field of machine translation with great optimism. One of the most famous examples of NLP progress was the program ELIZA (Weizenbaum, 1966), that had a fixed algorithm of rules on what phrases to use depending on the input from the user. Another example was SHRDLU (Winograd, 2004) - a program understanding natural language defined within a restricted domain with certain amount of objects, definitions and rules in the "world" of the domain. It was a precise and robust way to solve certain NLP problems, but not in many application fields or even varying cases in the same field.

Even though it gave a nice start for development of expert and recommender systems (Kazimierczak, 1990), afterwards the more complicated tasks required a more complex, better approach. The methodology was concentrating too much on the syntactic structure of the sentences, while it turned out the semantics and meaning behind the text was carrying the crucial importance. According to Su, Chiang, and Chang (1996), there are several major flaws in rule-based approach. Firstly, even though this method creates a comprehensive and compact system, it is very hard to upscale it. That is

mainly due to the costs of maintaining the large rule system without making it overly complicated or even worsen the effectiveness of the whole system in attempt of fixing some bad cases by adding more new rules. Furthermore, the rule-based approach was working very poorly with *"'ungrammatical' spoken prose and ... the highly telegraphic prose"* (Nadkarni, Ohno-Machado, & Chapman, 2011) of more formal technical texts. Such systems also translate badly to other application domains or languages, as Su et al. (1996) mentioned in their overview.

The solution was found in statistics and probability theory, under the assumption that *"human cognition is probabilistic and that language must therefore be probabilistic too since it is an integral part of cognition"* (Manning & Schütze, 1999). At the very beginning of the AI era scientists were full of ideas but were severely lacking computational power to bring them to life. In an attempt to minimize the drawbacks of both NLP techniques, some solution was offered in a shape of corpus-based statistic-oriented (CBSO) approach (Chen, Chang, Wang, & Su, 1991). Such a method implied that words can be clustered in some way having certain common properties essential for the processing. This technique allowed to use lesser training sets for the NLP systems, utilizing less computational power for the same tasks.

Still, the NLP problem quite often required larger corpora to work with and only the growth of the World Wide Web allowed to make that task much easier with all the amount of text flowing though it. The availability of bigger corpora caused the programmers. NLP research nowadays is often performed on the data sets collected from Twitter, Wikipedia, other social network sources. There is a general tendency pushing the NLP research towards Open Source Development, which can greatly decrease the costs of it and allow using and re-using such systems as flexible components in future work (Guerra, 2001). By the end of the second decade of 21st century the computational power has grown immensely, for example, giving an opportunity to have voice recognition systems not only on personal computers, but also on smartphones or even smart watches, making *"talking to your phone a commonplace activity, especially for young people"* (Hirschberg & Manning, 2015). Nevertheless, quite many aspects of natural languages such as ambiguity, irony, hidden meanings, etc., still prove to be an open topic challenging many researchers.

NLP consists of many various tasks with differing complexity. They can be subdivided in categories, depending on what part of the language they are dealing with. Some of those tasks are now well-defined and researched with main methodology adopted and wide-spread, while the others are still understudied and there is no common approach decided upon in the computer science society.

Further discussion in the context of this thesis goes more in detail about a particular NLP task - text summarization, which is in direct relation with the topic of this thesis. The aims of text summarization, the approaches and the advantages and disadvantages

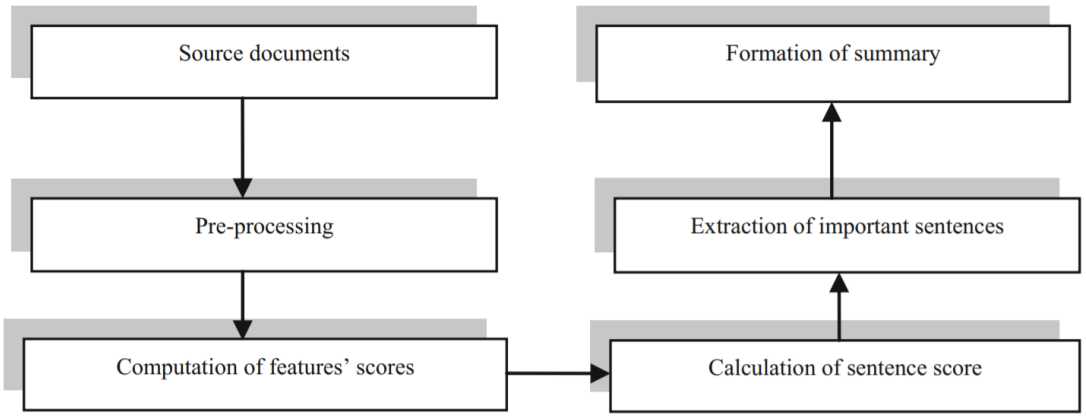

Figure 2.4: Block diagram of automatic extractive text summarization system by using statistical techniques (Gambhir & Gupta, 2017)

of them shall be described and looked into.

2.3 Text Summarization

> Text summarization is the process of distilling the most important information from a source to produce an abridged version for a particular user or task.
>
> Mani, Advances in Automatic Text Summarization

With the amount of information constantly and exponentially growing around us, the number of documents available online, the need of a tool being able to narrow it down, extracting only the most important part is getting more and more urgent (Gambhir & Gupta, 2017). Summarizing texts, articles and even conversations will give an opportunity to consume it faster and more efficiently. Four main requirements for a good summary was defined by Huang, He, Wei, and Li (2010) as following:

- **Information Significance**: only the important information from the original document should be added to the summary.
- **Information Coverage**: the extent of the information from the original document included in the summary should be maximized, however still being tightly connected to the significance requirement mentioned above.
- **Text Cohesion**: the summary should be grammatically correct and as readable as possible, not just a bunch of disconnected sentences and facts put together in an incomprehensible text.
- **Information redundancy**: the duplicate information from the original text is expected to be minimized, the summary should contain no factual repetition.

Text summarization techniques can be classified by the summary building method into *extractive* and *abstractive* approaches (Mani, 1999). Extractive summaries or extracts "*are produced by concatenating several sentences taken exactly as they appear in the materials being summarized*" (Nenkova & McKeown, 2011). Extracts appear to be very useful in the case when the user requires an overview on the document or a set of documents, without going through the whole content. It is supposed to provide the most important information picked and merged in one shorter summary - the workflow of one extractive approaches can be seen in Fig.2.4. The texts have to undergo some preprocessing stage if necessary, to "clean" it and prepare for actual summarization process, then the sentences are being scored and chosen to be extracted and added to the end summary by some criteria comprising of various features.

Speaking of preprocessing, one should not forget the importance of this step in the summarization pipeline. It is a step that many various NLP applications are relying on (Sunil, Jayan, & Bhadran, 2012) and it can greatly improve the end results if a correct preprocessing technique is chosen. There are numerous approaches aimed for that, such as:

- Sentence Segmentation - the text has to be split in sentences or utterances (in case of a transcript) to be able to assess separately the features of each like length, syntactical structure, importance, etc.
- Tokenization - segmentation of the text into even smaller units than sentences. For example, having a word "it's" a good tokenizer will split it into "it" and "is".
- Stemming and lemmatization - the process of bringing all inflected words of the same root to same form, either a canonical form (lemma) that actually belongs to the language or to a stem form reduced to the root. Mainly performed by suffix stripping and even by changing the word itself.
- Tagging - assigning such labels to words like part-of-speech or dependency tags for further utilizing such features in the pipeline. In many modern language programming tools such processes like tokenization and tagging are included and combined by default.
- Named Entity Recognition - another sort of labeling of the words, giving tags like "person", "place", "organization" to the ones that are recognized as the names of one of those entities.
- Stop-words removal - some words that occur fairly often but don't carry any significance to the NLP task have to be removed to decrease the fuzziness of the input data.
- Chunking - recognizing such structures within the sentence like noun chunks, verb phrases, etc.
- Word of phrase replacement - sometimes some words of phrases that essentially mean the same or close to being same should be unified. Processes like anaphora resolution, when the pronouns are replaced with the original noun they are representing, or replacement of the verbs with their hypernyms is used quite often for various NLP task.

- Other means of text normalization - removal of consecutive repetitions of unigrams, bigrams or trigrams, filtering out filler words or disfluences and other "clean-up" of the text to make it easier for an algorithm to work with.

However, summaries derived extractively are usually very different from a human-written summary (Yao, Wan, & Xiao, 2017). Due to grammar issues, the sentences might not be joined with each other through sentence connectors, that would sound natural and would surely appear in a summary written by a human. Nonetheless, in various cases such an approach still appears to be sufficient, providing the result good enough to stop on this method without further improvements. Abstractive summary, to the contrary to extractive, does not reuse the sentences or parts of them from the original text, but tries to reformulate and paraphrase them, creating new ones forming a summary. This task is more complex than extractive summarization, since it requires a semantic analysis of the text and it's abstract representation (Zhuge, 2015).

Another classification of summarization techniques is based on the aim of the summary: if it is supposed to give only the idea about what is the text about, it is regarded indicative; if the summary provides more information from the main text, it is called informative (Babar & Patil, 2015).

Summarization for humans is a straight-forward and fairly easy process: the document has to be read and understood, then the key points have to be picked out, reformulated and collected back in a coherent text of smaller volume than the source. However simple, the task may become time consuming with more text to summarize. This could be accelerated a lot if the computational speed of a computer could be applied. On the other hand, summarization is a complex task for a computer, it requires if not understanding of the whole text, then at least knowledge of the text structure.

At first the research in this field was concentrated on single-document summarization, trying to extract the main information from one single article, transcript, text, message or web-page. Certain techniques were proposed in the pioneer works in late 50s-60s: frequency of the words suggesting their importance (Luhn, 1958), sentence position and the occurrence of certain keywords as the main factor (Baxendale, 1958), or even the appearance of the words from the heading in the sentences being the defining reason to include it in the summary (Edmundson, 1969).

For a time being, those three main approaches were combined and used quite successfully for some tasks until novel algebraic and statistical methods started emerging. One of the first works among those was, for example, the system described by Kupiec, Pedersen, and Chen (1995), who suggested using a naïve-bayes classifier and a training set with texts with highlighted important segments in it to teach the system what parts can be valuable for extracting to the summary. Such an approach proved to be fairly fruitful, yielding 84% accuracy in case of the summaries being 25% the length of the original testing text, however, if the summary needed to be narrowed down,

the accuracy dropped. Various other methods were discovered and suggested, such as neural network approach (Yong, Abidin, & Chen, 2006), lexical chains (Barzilay & Elhadad, 1999), saliency criteria (Boguraev & Kennedy, 1999) and even some attempts to mimic human summarization techniques, such as sentence reduction (Jing, 2000) and "Cut and Paste" method (Jing & McKeown, 2000).

A need for big corpora by the end of 80s started growing more with overall adoption of statistic approaches. The Cognitive Science Laboratory at Princeton University started working on WordNet - first an annotated corpus of so-called "synsets", sets of synonyms and similar words grouped, purely for English language (University, 2010). This initiative later turned into forming of the Global WordNet Association creating other corpora in various languages (Association, n.d.). Most of the databanks are under open license, to propagate the usage of the corpora in research all over the world. Use of discourse structures and syntactical trees was introduced with creation of Penn Treebank (Marcus, Santorini, & Marcinkiewicz, 1993), a large corpus of over 4.5 million of English words with part-of-speech (POS) tagging. Carlson, Marcu, and Okurowski (2003) created a large corpus with discourse-level annotation for NLP research. The scientists started understanding that sharing such resources openly can greatly boost the research process, yielding to amazing results and achievements.

With the breakthroughs in computer science and improvement in computational power, the field of study was also expanded to multi-document summary. This was caused by the immense growth of information used and received in everyday life - whether that was e-mails collections, web sites catalogs or other digital libraries of large scale. One of the well-known techniques is TF-IDF - Term Frequency (Inverse Document Frequency) based method introduced by Salton (1989). It adopts the notion that important words are repeated more in various documents in the base that has to be summarized, however the system also excludes the very common words that are repeated constantly but bear no significant meaning for the summary. This was later adapted in various other works, evolving into TF-ISF (Inverse Sentence Frequency) (Gupta, Chauhan, Garg, Borude, & Krishnan, 2012) and other versions. Graph-based approaches also became quite popular in attempts to encode the textual or syntactical information from the documents into a versatile graph structure - like TextRank (Mihalcea & Tarau, 2004) with sentences as vertices and similarity score in between them, or the work of Zhang, Sun, and Zhou (2005) utilizing such properties of the graphs as centrality and network hubs. Other adoption from network calculation was the creation of LexRank (Erkan & Radev, 2004) and its further enhancements (Hariharan, Ramkumar, & Srinivasan, 2013).

The approaches discussed above were mainly applied to well-structured texts, such as scientific papers, reports, news, stories, etc. However, there is a specific subtask in summarization that deals specifically with dialogue and transcript summarization. The following section will concentrate closer on this topic and the discussion about the

approaches in this sub-field, as it is directly related to this thesis.

2.4 Spoken Language Summarization

> Consequently, automatically generated meeting summaries could be of great value to people and businesses alike by providing quick access to the essential content of past meetings.
>
> Wang and Cardie, *"Domain-independent abstract generation for focused meeting summarization"*

With time scientists started wondering whether the same summarization techniques that were discussed above are applicable to texts from other domains or of other styles. Dialog or meeting transcript summarization proved to be a tricky task for programmers - (Christensen, Gotoh, Kolluru, & Renals, 2003) described several experiments on applying already existing classic extractive summarization methods on speech recognition transcripts and concluded that more spontaneous speech provide less quality than organized structured text. Meeting transcripts are consisting of unstructured utterances with long-term semantic dependencies (Wang & Cardie, 2013). Such texts contain more grammatical and spelling errors, they are more noisy, thus producing a less readable and concise summary using extractive techniques (Liu & Liu, 2009; Murray, Carenini, & Ng, 2010). Still, there were many attempts to utilize extractive approach (Bui, Frampton, Dowding, & Peters, 2009; Riedhammer, Favre, & Hakkani-Tür, 2010; Xie, Liu, & Lin, 2008).

In some cases the result was enough to accomplish the task, however, it became apparent that to make a more coherent and sophisticated summary text, the sentences have to be adjusted and transformed. Here the research took different paths: sentence compression (Filippova, 2010; Jing & McKeown, 2000), template generation (Oya, Mehdad, Carenini, & Ng, 2014; Wang & Cardie, 2013) or sentence fusion (Banerjee, Mitra, & Sugiyama, 2015).

Some recent approaches in transcript summarization will be discussed in more detail now. The reader can find the brief overview collected in the Table 2.1. This review is mainly going to be concentrated on the latest works between 2010 and 2019 to better describe state of art and see the current picture of research in the field of dialogue and meeting summarization.

As it was mentioned above, scientists started turning away from extractive towards abstractive analysis, realizing that to produce a more readable and grammatically correct summary extractive approaches might not be enough. Nonetheless, by 2010 there were still some attempts in extractive summary that could be regarded successful.

Murray and Carenini (2008) suggested a system that tackles the conversation summarization as a classification task. They utilized a statistical classifier using various conversational structure features, such as sentence position, length, participant dominance, specific word usage, etc. Given all those features a logistic regression classifier was picking the best sentence to plug into the summary. The authors picked this specific sort of a classifier due to previous research (Cortes & Vapnik, 1995) proving that even though the quality of the results of a logistic regression classifier and a support vector machine (SVM) was fairly equal, the SVMs take way longer to train than logistic regression classifiers. The evaluation of the system showed that some of the features turned out to be more useful than the others in different application domains. The authors claim that the system is robust even in noisy datasets and provides still useful summary information about meeting or email conversation in a very short time, even giving a prospect of being extended to other domains.

Another extractive approach was described by Bui et al. (2009) - this time steering more into so-called "focused summarization", which *"in contrast to summaries of a meeting as a whole, they refer to summaries of a specific aspect of a meeting, such as the DECISIONS reached, PROBLEMS discussed, PROGRESS made or ACTION ITEMS that emerged"* (Wang & Cardie, 2013). This particular work concentrated on classifying sentences into different dialogue acts to pick up the ones related to decision making. Such a procedure is executed using Directed Graphical Models (DGM) to model sequences and dependencies in the conversation structure. The system could detect three main decision dialogue acts (DDA): issue, resolution, and agreement. After that, the algorithm was following two rules in decision region selection:

- The decision discussion region begins with an issue DDA.
- There has to be at least one issue and one resolution DDA in the region.

Such a region was picked for decision summary generation. Agreement DDA normally didn't contain any essential information regarding the problem, thus it was omitted from the summary. An SVM regression model was picking the best short fragment that was most likely to match the gold-standard extractive summary. Ultimately, DGM when using non-lexical features proved to outperform hierarchical SVM classification suggested before by Fernández, Frampton, Ehlen, Purver, and Peters (2008). The authors experimented with different feature sets and data, drawing conclusions that could lead to future work and improvements.

In the meantime, abstractive methods were rising among the community. Murray et al. (2010) proposed document interpretation based on general conversation ontology with "message" generation - small summaries over multiple sentences - and further picking of the most informative messages. Suggested ontologies are describing not only high-level entities like Participant, Utterance or DialogueAct, but also subclasses and properties. This way, for example, ProjectManager is included in Participant or DialogueAct has various subclasses corresponding to different phenomena: decisions,

actions, problems, etc. Sentences are classified by a pre-trained system to map them to such an ontology description. The process doesn't stop at merely classifying sentences - the authors attempted to make a system that can recognize bigger patterns in the conversation, they called them "messages". An opening or closing of the meeting can be classified as a message, or repeated agreement or disagreement, decision making process or problem discussion. An integer linear programming (ILP) then selects the most informative messages among all the detected ones, and using all the information from the ontology representation with the means of simpleNLG[1] a sentence for each message. With a schema-based approach the planning of the end article is performed and the summary is assembled. A general downside to such an approach is the requirement for pre-training labeled datasets. The results showed that this technique outperforms human-written extracts with better readability, coherence and usefulness scores, but still loses to abstracts created by people.

A full summarization pipeline was suggested by Mehdad, Carenini, Tompa, and Ng (2013) being similar to the approach of Murray et al. (2010) with changes to the content selection step and different technique applied to the summary generation phase. Unlike Murray et al. (2010), the authors used lighter approach to annotation, having only links between sentences in a human abstract and the sentences in the original text. Sentences were classified pairwise whether they can be abstracted together by a new sentence and a graph was build with sentences as nodes and edges as those classified connections. Afterwards, communities were detected inside the graph calculating betweenness of the nodes, single sentences with no connections represented their own singleton communities. To avoid redundancy and repetition, an entailment graph was created with a supervised method for each community, recognizing important and new information among the sentences. Normally, the nodes with more outgoing entailment relations and the roots of longer entailment chains are being regarded essential and informational. Finally, sentence fusion was performed with the help of a Word Graph based on the method proposed by Filippova (2010), merging identical words or synonyms, replacing some words with their hypernyms. Several sequences could be generated from the graph following the possible paths (see Fig.2.5), which later had to be ranked based on readability, informativeness and other scores to pick the best version to include in the summary. Certain drawbacks were detected in such an approach after the experiment testing. Firstly, since the generated sentences are still based on the sentences directly from the transcript, it's following the informal style of the original text, while human-created abstracts are translated to a proper formal writing style. Secondly, the subjectivity of human-written abstracts also distorts the way the program then tries to generate the summary. Lastly, since the speaker information is not taken into consideration, the summary does not give any participant description or naming, as the human-produced abstracts normally do. Also, it became apparent that such texts like meeting transcripts contain various grammatical and spelling errors and need to be normalized and pre-processed to improve some results. On the positive

[1] https://github.com/simplenlg/simplenlg

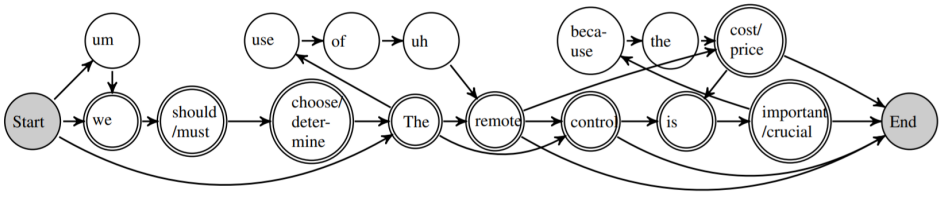

Figure 2.5: Word graph generated for a sentence utilized for sentence fusion. The arrows show possible fusion paths, double-bordered nodes contain merged words.(Mehdad, Carenini, Tompa, & Ng, 2013)

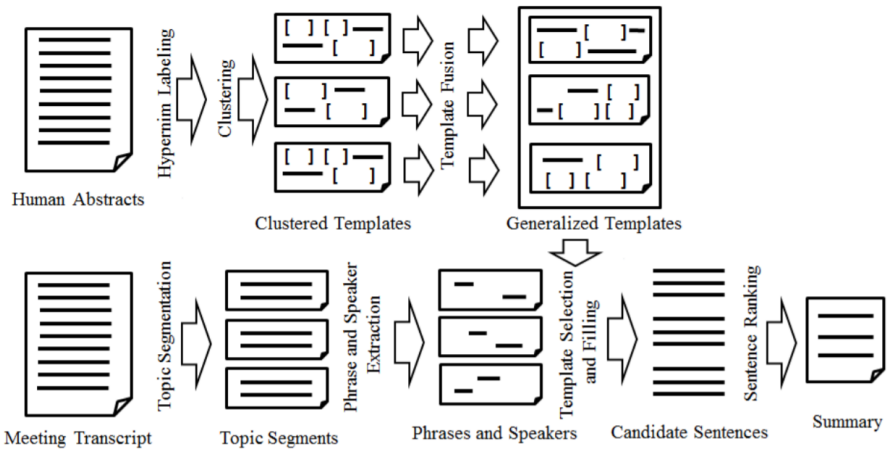

Figure 2.6: Two-component meeting summarization framework presented by Oya, Mehdad, Carenini, and Ng (2014)

side, the system proved to be capable of generating longer sentences while still keeping them relatively grammatically correct, which is competitive to previous word graph based approaches generating shorter sentences, and the informativeness of the summaries in general is higher than other meeting summarization models suggested before.

Oya et al. (2014) followed with a template-based approach to meeting summarization. This system is also using the word graph method, however, this time for template generation. The whole framework consists of two components (see Fig.2.6) - offline template generation and online summary production. The template generation module was designed in such a way so it could possibly create the templates general enough, however also quite specific, so each template only accepts certain fillers. Sentences with active root verbs are collected from human-written abstracts, noun phases replaced with hypernyms, and after classification this blanks are fused using a word graph into the final templates. For the summarization component, topic segmentation was applied according to the method proposed by Galley, McKeown, Fosler-Lussier, and Jing (2003) with post-processing extensions. Salient sentences were extracted based on the frequency of each word in the fragment, the same hypernym replacement conducted on noun phrases. Each template was linked to a community of sentences

from the training data, so in the search for a better summarizing sentence for an actual community in the current text the most similar training community had to be picked. Finally, the multitude of sentences generated was ranked based on such criteria as fluency, coverage, etc. and the best ones were chosen to build up the summary. This work brought such a template generation approach as a novelty together with the template selection technique, and according to the testing, the summaries were outperforming the human-written extracts as well as the results from current works.

In the field of focused summaries, Mehdad, Carenini, and Ng (2014) proposed phrasal query-based approach to such task, to directly address the needs of the user for any specific information needed from the document. The utterances were being extracted following two criteria: containing the essence of the text and the answer to the user query. The authors decided to utilize the concept signature and query terms - with log-likelihood ratio for the first case and WordNet synsets for the second. Those utterances were scored by maximizing the coverage, some of them removed through an entailment graph afterwards to avoid redundancy. The rest of the procedure is similar to some approaches already discussed above - clustering, finding the best path over the word graph based on a ranking technique. As a result, the system proved to be correctly producing query-based summaries with good grammatical scores from both automated and manual evaluation.

Another graph-based approach was suggested by Banerjee et al. (2015). It is another example of graph sentence fusion per each topic fragment, when the best summary sentences are chosen by finding the best path on a word-graph. However, unlike the graphs described in Mehdad et al. (2013), in this case the authors applied dependency parsing to build the connections between words. Moreover, some reference issues were attempted to be solved, when some entity is named in one sentence and referred by a pronoun in the following ones - that creates problems for sentence fusion, so such noun phrases have to be unified by anaphora resolution: replacing all the pronouns by the original noun. ILP approach was used for path selection. The results have shown that anaphora resolution indeed improves the evaluation scores and the produced summaries outperform extractive summary model that served as a baseline.

Markov Decision Process (MDP) was used as a summarization technique by Murray (2015). Firstly, the already known community detection had to be applied and several different techniques were used for comparison: a supervised logistic regression, unsupervised k-mean clustering and human gold-standard sentence communities. The summarization MDP state structure is illustrated in Fig.2.7 - the states are representing unique word types occurring in every cluster. The sequence of words is generated in between the START and STOP states, producing a possible sentence for a cluster summary. Value Iteration allowed to pick the best possible word at every step and state thinning resolved the issue of word repetition in a sentence. Moreover, the average length of a produced sentence can be regulated by determining the number of time-

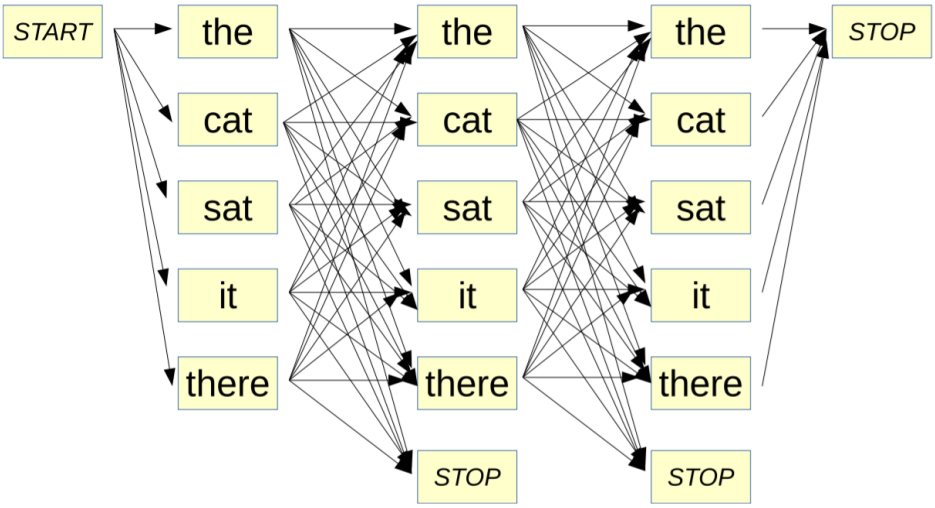

Figure 2.7: An example of Markov Decision Process state structure for a simple sentence (Murray, 2015).

steps, the so-called ”horizon”. In the end, the summary quote often consisted of some sentences being completely identical to the ones in the original text, some of them were shortened, and some of them represented a fusion of different ones. In conclusions the authors were discussing the idea of combining the MDP approach with top-down template filling, due to the MDP being flexible with possible constrains on some fixed patterns from the templates. Unfortunately, the sentence fusion performed by the MDP quite often lead to the sentences being not grammatically incorrect or nonsensical.

As it can be seen, community detection is occurring very often in such works, either directly for summary generation or for template generation step. Singla, Stepanov, Bayer, Carenini, and Riccardi (2017) discussed various heuristics for such an operation: taking the whole text as a community for each sentence, 4 closest turns with respect to cosine similarity between the summary sentence and the conversation sentences, 4 closest turns but after replacing the verbs using synsets and 4 closest turns based on similarity with average word embedding vectors. As a result, the last technique turned out to be more effective than the rest, however, the system was tested in two languages - English and Italian, and the Italian version was showing lesser difference in the performance of all four approaches. This can be possibly explained with smaller train data available for the Italian language, which decreases the precision of the system.

Among the neural network approaches was the work suggested by See, Liu, and Manning (2017). The authors tried to address the main shortcomings of ordinary sequence-to-sequence approaches: incorrect factual detail representation and repetitiveness. The first issue was suggested to be tackled with pointing methods, which would allow more accurate reproduction of information by copying some words directly from the original text. The second problem was attempted to be solved involving coverage

monitoring to keep track of what has been already summarized. The pointer-generator network consisted of an encoder and a decoder and was deciding the probability of a word at each step either being generated or copied from the text. For the coverage mechanism an adapted version of the approach by Tu, Lu, Liu, Liu, and Li (2016) was used, helping the decision making at each step with a reminder of the decisions already taken previously. This way repeated attention of the network is prevented and repetition of the factual information is minimized. The system can be considered partially abstractive, because of the copying of the information from the original text, however the evaluation has shown that in the end it still even outperforms many of the state-of-art abstractive solutions, and in general there is a perspective of encouraging the network to write more abstractively but still retaining the accuracy of the pointer technique.

Since already many approaches were suggested and shown relatively good results, many further works tried to combine them somehow in an attempt to boost the performance even more. Shang et al. (2018) experimented with combining already described community detection technique using TF-IDF vector space and multi-sentence utilizing the word-graph representation. Summary sentences are generated, ranked by several values like coverage or fluency, and then by maximizing a submodular and monotone non-decreasing objective function the set of summary sentences is reduced to a desired summary length and redundancy and off-topic content is being decreased. A benefit of this work is that the approach is fully unsupervised, meaning no annotation or pre-training is needed. The input is just pure original text without any metadata and only a language tool with a model, POS-tagger, word vectors and stopword lists is required. This makes the system very versatile and able to work out-of-the-box with different languages given a language model, and it is not domain-dependent as well.

Among the latest works Ganesh and Dingliwal (2019) presented a new approach yet again being a fusion of methods from previous papers: sequence-tagging the transcript and modeling a discourse structure with application of an attention-based network to it afterwards to generate the summary. The idea of using discourse structure of the transcript is not new, the authors were in some way following the example of Stone, Stojnic, and Lepore (2013), however, drastically simplifying it due to modeling being the only purpose of creating such a structure. The discourse structure data and lexical information is used to remove abandoned and unfinished sentences, pauses, non-verbal cues, etc. The coverage-based pointer network approach is borrowed from See et al. (2017) without any additions, only with adaptions to newer versions of frameworks used in the pipeline. The result evaluation showed the abstractive properties of the end summary hence improving the readability score.

While some of the papers returned to discourse structure application, the others revisited encoder-decoder neural network approaches like Zhao et al. (2019). The authors employed the hierarchical encoder technique proposed by Li, Luong, and Jurafsky

(2015) in an attempt to model long-term semantic dependencies in a conversation. To learn the semantic representation of the meeting transcript, an adaptive encoder inspired by binary neurons is applied to the texts. Utterance-level Long short-term memory (LSTM) networks help fragmenting conversational topics in the text. Afterwards a reinforced decoder network based on segment-level LSTM networks is used to generate summaries of the topic fragments - given the semantic representation, the decoder predicts the next word in the summary on each step. Reinforcement learning had to be applied to pre-train the decoder and optimize the network. The resulting summaries show striking fluency and appear rather natural, still retaining coverage of necessary factual information. The outcome can already be compared to human-produced abstracts, which is essentially really close the aim that computer scientists have been aspiring to achieve already for several decades.

As a general picture, the tendency goes more and more towards abstractive summarization nowadays, when the researchers trying now not only just represent correct data obtained from the original text, but also make the final text sound as natural as possible, making it look like it has been written by a human and not by an algorithm. Furthermore, there are numerous attempts in focused summarization, which narrows down the amount of information the user gets from the transcript even more, concentrating only on personalized shorter summary and making the abridged version more precise and effective.

Table 2.1: Transcript Summarization Techniques

Work	**Type**	**Methodology**	**What's new?**
Murray and Carenini (2008)	extractive	Machine learning classification using conversational features to detect saliency	The system is not domain-restricted and outperforms state-of-the-art domain-specific summarization tools.
Bui et al. (2009)	extractive	Various dialogue act classification to detect the phrases that concern decision making and outcome and dependencies between the phrases	Directed Graphical Model used to describe sequences and dependencies, use of similarity measures to improve sentence selection
Murray et al. (2010)	abstractive	Input sentence ontology mapping based on the set of features relating to conversational structure and sentence-level phenomena, abstract generation over multiple sentences, most informative abstracts selection, final text generation based on the picked abstracts	Improved readability, coherence and informativity, fully automatic summarizer

Mehdad et al. (2013)	abstractive	Entailment graph on communities of clustered sentences, word graph with ranking for selecting the best path on the graph	Abstractive summary generation utilizing word graph model for sentence fusion, utilization of semantics in textual entailment graphs, method is not domain-specific due to minimal syntactic information usage.
Oya et al. (2014)	abstractive	Multi-sentence fusion and lexico-semantic information for template generation, word graph, utterance extraction based on topic segmentation	Novel approach in template generation. The generated summaries are generally preferred by the participants of the user study to extractive ones and other state-of-the-art meeting summarization systems.
Mehdad et al. (2014)	abstractive	Ranking and extracting utterances based on content and phrasal query, clustering of extracted sentences by similarity, word graph application for aggregation with ranking for final summary sentences selection	Query-based focused summarization, concentrating on the required factual information, high grammaticality of end summary.
Banerjee et al. (2015)	abstractive	One sentence summary generation per topic segment by fusing the sentences each one	Robust approach for noisy data (including disfluences, etc.) outperforming extractive approaches.
Murray (2015)	abstractive	Summarization problem as MDP for community detection among transcript sentences	MDP proved to be superior to extractive approaches, however synthesized sentences are ungrammatical and nonsensical. Application to other domains possible.

Singla et al. (2017)	abstractive	Template generation applying slot labeling, summary clustering and fusion, automatic community creation using cosine similarity for template selection, topic classification using a lexical cohesion-based domain-independent discourse segmenter	Testing different cosine similarity heuristics by calculating on different levels: raw text, text with replaced verbs and average word embedding similarity; testing on English and Italian corpora
See et al. (2017)	abstractive	Neural sequence-to-sequence model augmented with a hybrid pointer-generator network and coverage model to avoid repetition	A methodology suggesting to fix two downsides to previous sequence-to-sequence approaches - correct factual information reproduction and repetitiveness.
Shang et al. (2018)	abstractive	Community detection for sentence clustering, single summary sentence generation per topic using Multi-Sentence Compression Graph, summary sentence selection by maximization of a custom submodular quality function under a budget constraint	A combination of several previous approaches in an attempt to utilize their strengths, fully unsupervised framework - the system does not rely on any annotations or training sets and also not English-specific.
Ganesh and Dingliwal (2019)	abstractive	Attention-based pointer network using discourse relations in the dialogue using sequence tagging	Use of lexical information to remove pauses, abandoned sentences, nonverbal cues etc. and replace acknowledgments, appreciations, agreements etc. for a more informative summary
Zhao et al. (2019)	abstractive	Neural network approach - hierarchical neural encoder based on adaptive recurrent networks to learn the semantic representation of meeting conversation with and decoder based on segment-level LSTM networks to generate the summary	Adaptive segmental encoding introduced

2.5 Summary

In this chapter various topics related to this project were brought up, starting with news, media, journalism and introducing a discussion on the current state of this industry and the influence of the rising computational journalism on the reporters, readers and society in general. After the debate on the pros and cons of such a change in the news production, the main point of analysis switches more to the technical part of this thesis - mainly, explaining about NLP aspects and giving a short overview on its history, with later narrowing it down to the subfield of NLP, text summarization. Some classification of summarization tasks is explained with some examples given, and then a detailed analysis concentrates later on the issue directly relating to the task of this thesis - conversation summarization. An in-depth review is provided on the latest achievements and novelties in this field, describing the techniques emerging and observing the results of its application. Such a review can greatly help with choosing the right approach for the task given to this thesis with the right assessment of the circumstances given and options available.

3 AI For Reporters

Ai For Reporters is a part of the bigger initiative originating from California Polytechnic State University called Digital Democracy. The whole idea behind it is to make politics more accessible, more transparent and available to common citizen, the press and anybody else interested in the political development of the United States. To have a real strong democracy in the country, the average citizen needs to be well informed. Moreover, the information that people consume should ideally be without bias, representing only facts, dates, numbers and events that can be easily proven and traced back to its origin. Luckily, Digital Democracy has been building up such a base with facts and texts for years (Blakeslee et al., 2015; Budhwar, Kuboi, Dekhtyar, & Khosmood, 2018), and the AI For Reporters project is aiming to utilize it to the maximum. The main source of data for the project is the hearing transcript database within the Digital Democracy initiative created by human-assisted annotation methods (Ruprechter, Khosmood, Kuboi, Dekhtyar, & Gütl, 2018).

In this chapter, Section 3.1 outlines the main requirements and goals of the project, describing the desired ideal result of its work and what was set originally to be achieved with it. After which, in Section 3.2, the main concepts adopted are being explained, with main approaches and methodology documented within the section. Further on, in Section 3.3, all the libraries and frameworks used in the project are being described, explaining their meaning and place in AI For Reporters structure.

This thesis does not extend the Digital Democracy initiative, AI For Reporters is designed only to purely make use of the database already created by the Digital Democracy team.

3.1 Requirements

Defining the requirements is a crucial part for any development process, which gives clarity to the goals and steers the project in the right direction from the start. The requirements can be split into functional and non-functional, with both being equally important for the flow of the development process (Capilla, Babar, & Pastor, 2012).

AI For Reporters is a data-driven template-based summarization system. The following requirements can be listed as functional:

- The facts supplying the program are either queried from the Digital Democracy database directly, or mined from the transcript texts of the legislator hearings

stored in the database.
- Given a hearing ID, the system should fetch all the data connected to it with a query and begin processing and fact extraction, with which the article text templates are later to be filled and arranged in a grammatically correct and readable report.
- Each fact has it's own source for transparency reasons which is marked in the article with a footnote, allowing the reader to see the background of each statement and understand how it appeared in the text.
- The system has to be flexible enough to allow an addition of new types of facts to be mined or exclusion of the ones that represent no interest to the end user. It should be designed in such a way that external contributors could still add their own fact mining blocks without knowing how exactly the whole system works. Ideally, there should be minimum connection points that they would have to interact with to expand the summarizer functionality.

Among the non-functional requirements certain qualities can be defined that are expected to be present in the project:

- The fact extraction system must have high precision to be credible, meaning that the summarization has to be robust and has to have low tolerance of false results and incorrect facts represented by it.
- The system must provide a cohesive and readable end summary text as well as a collection of all the facts gathered, all the assets, pull-quotes, links and footnotes in one single file that can be provided to the end user.
- The execution time of the system should be short enough to be able to provide quick summaries for the end user on request and upon the new data emerging in the database. This can ensure that the news provided in the summary articles is topical and of current interest, keeping up to date with quickly evolving events nowadays. The execution of the program should not take longer than it would take a human reporter to create an abstract from the hearing.
- The summary should have some abstractive properties to it and utilize not only the facts already existing in the database, but also utilize the text of the transcript itself, performing some NLP analysis and mining the facts from it directly that otherwise could be only discovered by watching the recording itself by a human.

3.2 Concept Description

As it has been already mentioned, one of the main aims of this work was to utilize the amount of data in the Digital Democracy project to the most of it. Meaning, not only work with the metadata available, but also try to extract the facts that could be interesting to the reader directly from the transcript of the hearing. After observing the hearing videos and reading through various transcripts, it appeared to be clear, that some patterns can be recognized and some data can be extracted from it.

This inspired the adoption of so-called ”phenom” approach - extraction of the key highlights from the transcript text and putting them all in a collection of facts. For each fact at least one template has been manually created, which are later either filled in and added to the final product text if the corresponding fact has been successfully mined, or is simply discarded if there is not enough facts in the collection to fill in the template.

Such an approach requires a close study of various transcripts and hearing, a lot of observation to track common features and patterns among the texts. Every hearing transcript may contain at least one or two facts that could be interesting to the summary reader and be worth extracting and adding to the resulting text. Digital Democracy database contains a multitude of videos to first examine how the legislation hearing proceeds, which parts of it is just a necessary agenda and which are special outliers, extraordinary happenings and events that might draw attention of the reader as if they would be present in the hearing itself. Each of such events can be represented as a phenom and produce a sentence or two for the end summary article.

Another important feature of such a concept is the ability to expand. Considering that the phenoms are unified in a specific generic way, it should cause no trouble to add new ones to the system if need arises, without any crucial changes in the architecture. Furthermore, other people working on the project could be also engaged in creating their own phenoms, without any in-depth knowledge of the code. Only knowing the entry and end-points for a phenom would be required to create and extension.

If any phenom can be called in the same way, some intelligent algorithm can be derived for the dynamic building of the article, such as a partial order planner or a similar technique. This creates some certain randomness in the article construction, and with the addition of multiple possible templates per phenom, the texts can vary a lot and not sound so ”robotic” and bland and be closer to human-written abstracts.

3.3 Tools and Frameworks

AI For Reporters is developed in Python3 (Van Rossum & Drake, 2009), mainly due to the various packages for natural language processing and working with text available in this programming language. To be exact, two different Python packages were used for NLP tasks:

- SpaCy (Honnibal & Montani, 2017)
- Natural Language ToolKit (Bird et al., 2009)

Both tools are capable of parsing, tokenization, lemmatization and dependency tree building - all of the processes necessary for proper text mining. In tokenization SpaCy has proven to deliver better results, however in tasks like single-word lemmatization NLTK performs good enough and is more lightweight than SpaCy with the NLP pipe

call so it doesn't slow down the process as much. Pyton library re[1] with tools for regular expressions is also invaluable for text processing and preprocessing in some parts of the task.

Scikit Learn (Pedregosa et al., 2011) together with numpy (Oliphant, 2006) serves nicely for classification purposes, providing useful built in classes and methods to train and use different classifiers.

For the database calls a library called MySQLdb[2] is utilized, establishing connection to the Digital Democracy database and retrieving the needed data, whether it is some information about a speaker or the hearing transcript. Two main data structure libraries are used for storage and data collection - Python package for JavaScript Object Notation (JSON)[3] and Pandas (The Pandas Development Team, 2020). The transcripts are being stored in a Pandas DataFrame (McKinney, 2010) as a table containing fields with information about the speakers, the hearing itself, the utterances, etc. The DataFrame has a very versatile structure that allows accessing by indices, column or row names, slicing, joining and other manipulations. Moreover, DataFrame allows transformation to various different formats, such as Excel sheets, JSON objects and strings, arrays, etc. JSON structures are used for building up the output and presenting it to the end user in a an adaptable and functional way. For the templates Python String Template[4] class was chosen, for convenient filling of the sentences with collected facts. Template sentences are being pulled from a shared Google spreadsheet by the means of a Python library urllib[5].

Some phenoms that extract the whole sentence, for example, a pull quote, require some fine ranking system for that, and one of the most important criteria for a quote is readability. There are certain techniques and scores that can be used for such a check - Automated Readability Index (Senter & Smith, 1967), Flesch-Kincaid formula (Kincaid, Fishburne Jr, Rogers, & Chissom, 1975), SMOG grading (Mc Laughlin, 1969), etc. Python library Textstat[6] implements various readability rating techniques and allows an easy application of such formulas to texts and sentences.

3.4 Summary

In this Chapter such an important step for software development as requirements definition has been discussed. Furthermore, a description of the newly introduced concept utilized for this project is given, as well as an overview of the tools chosen

[1] https://docs.python.org/3/library/re.html
[2] https://mysqlclient.readthedocs.io/user_guide.html
[3] https://docs.python.org/3/library/json.html
[4] https://docs.python.org/2.4/lib/node109.html
[5] https://docs.python.org/3/library/urllib.html
[6] https://github.com/shivam5992/textstat

for the development of AI For Reporters summarization tool. In the next Chapter some more detailed description of the architecture and the technical details of the development process will be given, explaining more about the concepts and steps of the project and also describing how they were actually implemented.

4 Development

In the following chapter the overview on the architecture of the project will be given, as well as the implementation process with the technical description of different components will be described. Section 4.1 demonstrates the pipeline and the workflow of the program, giving an overview on it's components. Certain modules that are supposed to be a part of the AI For Reporters project and be included in its pipeline will only be shortly brought up without going into any detail about the implementation of those components.

Section 4.1 outlines the architecture of the project, telling about all the components and important steps of the workflow. Section 4.2 then follows up with technical details about the implementation of those components. The application of the tools introduced in Section 3.3 is described as well as some development solutions.

4.1 Architecture

The program architecture (see Fig.4.1) is designed as following: the hearing identification number is provided by the newsworthiness ranking module, after which the requested hearing transcript is pulled from the Digital Democracy database with an SQL call. The newsworthiness selection mechanism is an external project currently in progress and will not be discussed in this thesis. Then the paragraph classification is performed, splitting the transcript into classified fragments using the predicted labels. After some preprocessing and separation of the fragments classified into categories earlier, the program is ready to start extracting facts from the transcript text. There is a collection of various classes created that can return one or more facts derived from the text. They have a system of pre- and postconditions and are being called by a partial order planner. Each of those methods has at least one corresponding template stored in the template bank, which are getting filled on execution of the methods of such a class. Some of the templates might require additional information from the database, thus API calls are also utilized at this step. After the execution of all possible fact extractors is finished, the final article is assembled from the filled templates according to the created plan, and all the data collected with various assets, headline, article text, etc. is written to a JSON file that can be delivered to the end user or demonstrated on the Ai For Reporters web page.

Among the approaches discussed in Section 2.4 many work utilized template-based summary generation, either with hand-written templates or automatically generated

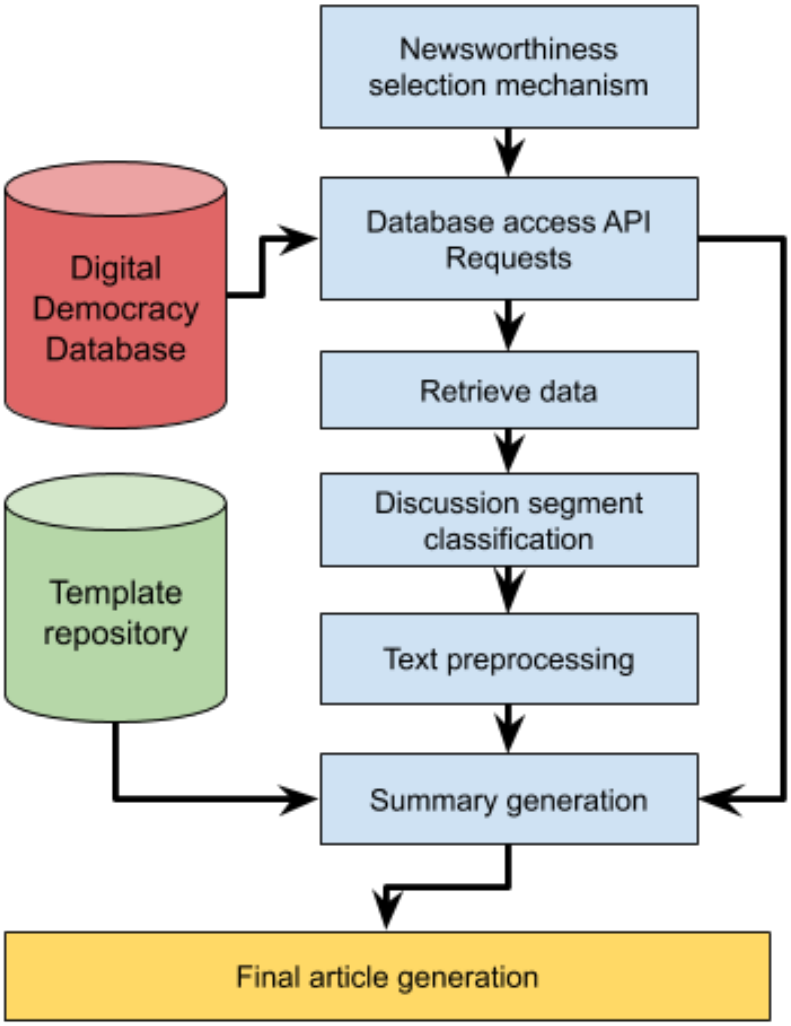

Figure 4.1: The workflow diagram of the AI For Reporters project

	C	E	G	H	I	J	L	M	N	O	P	Q	R
1	alignment	did	first	hid	last	pid	state	text	time	type	uid	vid	vid_file_id
2	Indetermir	26114	Travis	52774	Allen	43	CA	SB1322 went into effect January 1st. Under this law, child prostitution is now leg	30	Discussion	13573646	27651	48cc14638aa8a34cdba8b
3	Indetermir	26114	Reginald	52774	Jones-Sawy	87	CA	Thank you. Any witnesses in support?	158	Discussion	13573654	27651	48cc14638aa8a34cdba8b
4	Indetermir	26114	Travis	52774	Allen	43	CA	Today, I brought two individuals. The first I'd like to introduce is Detective Anthor	160	Discussion	14681710	27651	48cc14638aa8a34cdba8b
5	For	26114	Tony	52774	Guerrero	112565	CA	Hello, my name is Detective Tony Guerrero and I've been working fighting human	172	Discussion	13573656	27651	48cc14638aa8a34cdba8b
6	Indetermir	26114	Reginald	52774	Jones-Sawy	87	CA	Thank you. Next witness.	363	Discussion	13573666	27651	48cc14638aa8a34cdba8b
7	Indetermir	26114	Travis	52774	Allen	43	CA	Thank you. Next, I have Jason Parker, Chief Investigator for the Sutter County Dis	367	Discussion	13573667	27651	48cc14638aa8a34cdba8b
8	For	26114	Jason	52774	Parker	112568	CA	So, last year, we started working with some nonprofit groups. Operation Undergr	375	Discussion	13573668	27651	48cc14638aa8a34cdba8b
9	Indetermir	26114	Reginald	52774	Jones-Sawy	87	CA	Thank you. Are there any other witnesses in support? Are there any witnesses in	567	Discussion	13573679	27651	48cc14638aa8a34cdba8b
10	Against	26114	Jodie	52774	Langs	18510	CA	Good morning. Thank you. My name is Jodie Langs. I'm with Westcoast Children	602	Discussion	13573681	27651	48cc14638aa8a34cdba8b

Figure 4.2: A fragment of a bill discussion data table fetched from the Digital Democracy database.

ones. Such a methodology allows a more robust system producing grammatically correct sentences with possibly lower coverage but better precision, which follows one of the requirements brought up in Section 3.1. Human-written templates will also in the future allow other project contributors to easily add either other versions of templates for already existing phenoms, or add up new ones for newly-created additional phenoms. This decision goes also in accordance with other requirement about the system being easily extendable, which was outlined in Section 3.1.

4.2 Implementation

The following Subsections explain the principles and mechanisms of various parts of the project, giving some examples and technical details for a better understanding of the work conducted in this thesis.

4.2.1 Data Structure and Storage

The database API uses various SQL requests via Python library MySQLdb to retrieve certain data from it. All sorts of database requests created over the course of this project were stored in a common Python script file `db_queries.py` for convenient reusing at any other point within the program. Inside this script a Database Class is implemented with all the information about credentials and initiating the access. Each method of that class contains a specific database query addressing a certain need for any data available within Digital Democracy.

One of the main and biggest requests is the initial data retrieval: knowing a discussion identification number (later **did**), the program sends a big joined request over the database tables fetching all the data required connected to that **did** and stores it in a Pandas DataFrame (see Fig.4.2). The columns of this DataFrame contain such information as **did**, speaker id (**pid**), hearing id (**hid**), utterance text, first and last name of the speaker, alignment of the speaker, etc. Each row represents all of this data per one utterance. For debugging and logging reasons such a DataFrame can be written to a Microsoft Excel sheet with Pandas library method `pandas.DataFrame.to_excel`. All the training data examples discussed further were also stored and labeled in Excel tablesheets.

4.2.2 Paragraph Classification

To approach the summarization task in this particular case, the decision was made to split the hearing in certain segments, representing particular event happening throughout the meeting. Various approaches discussed earlier in Chapter 2.4 adopt a technique of splitting the text in fragments and then summarizing each part separately, producing one or more sentences based on the information extracted from it. Such an approach seemed promising for the type of the text this project tries to summarize mainly for two reasons - firstly, a legislative proceeding is indeed mainly well-structured and has certain steps on the agenda for the legislators to go through, and secondly, this agenda stays the same in most of the cases.

After investigating various transcripts of the meeting, the following segment types were defined:

- *Organizational* - reading the agenda, presenting some members, announcing first some information unrelated to the bill discussion
- *Intro* - the Chair or the Clerk reads the number of the bill and calls out the person to make the introduce the bill to the audience, the presenter talks about the bill and in the end encourages the audience to vote in favor
- *Testimony and questions* - the invited experts and the public is invited to testify for or against the bill, the audience is asking any questions related, optionally a motion on the bill is proposed and seconded
- *Voting* - the voting on the motion is announced, the votes are gathered and read out by the Clerk
- *Closure* - The meeting is announced to be adjourned or the next bill presentation is called out

These paragraphs also often contain some procedural language to mark the beginning or the end of each segment, so it was decided that a program can be taught to recognize such words and phrases and detect those borders. Thus, such a task can be tackled as a classification problem. Afterwards, each fragment can be separately analyzed and summarized on its own.

Training Dataset

To prepare any classifier a training set was needed, and since the problem being so case specific, there was no other way but manually label some data and train the classifier on it. 40+ actual meeting transcripts of various length from 10-15 up to 2000 utterances were taken as a test data set. Human annotators had to read through these texts, labeling the beginning and the end of each specific fragment within a hearing. Such an approach was aimed to help to teach the classifier to distinguish the border, where one fragment ends and another begins. Integer labels from 1 to 5 were assigned to the categories and label 0 was representing a non-border utterance within the fragment. An example of such a manually labeled hearing can be seen in Fig.4.3. One can see the

	A	B	C	D	E	F
	utterance	pid	positive	text	Paragraph label	Secondary label
1		92	1	Everybody turn off their cell phones if they want to bother us while we spend the next 3 to 4 minutes in this committee. And lets take our roll. We have 4 measures, all on consent items. And all the items on our agenda are on consent. So we'll take the roll and then we have a motion on the consent calendar.	1	
2		2998	1	Senators Beall? Here. Beall present, Cannella? Here. Cannella present. Allen? Bates? Gaines? Galgiani? Leyva? Here. Leyva present. McGuire? Here. McGuire present. Mendoza? Here. Mendoza present. Roth? Here. Roth, present. Wieckowski? Here. Wieckowski present.	0	
3		92	1	Okay these consent calendar items are ACR 58 by various Assembly Members, Williams, ACR 63, Maienschein, and ACR 65, Brough, ACR 78, Salas, and if there's no discussion on these items, we'll have a motion for the approval of the consent calendar. The motion's to approve this consent calendar, and we'll take a vote.	2	4
4		2998	1	On the consent calendar, Senators Beall? Aye. Beall aye, Cannella? Aye. Cannella aye. Allen? Bates? Gaines? Galgiani? Leyva? Aye. Leyva aye. McGuire? Aye. McGuire aye. Mendoza? Aye. Mendoza aye. Roth? Aye. Roth aye. Wieckowski? Aye. Wieckowski aye.	4	
5		92	1	7? We're gonna put that on call, we have 7 votes in favor. So we'll wait for other members to come and record their votes, and as soon as they come and vote, we will adjourn the committee. So we're waiting on 4 Senators, and we'll wait for them and record their votes and adjourn.	0	
6		2998	1	Thank you Senator Beall.	0	
7		92	1	Thank you.	4	
8		92	1	Consent calendar.	2	
9		2998	1	Senators Allen? Bates? Gaines? Galgiani? Galgiani aye.	4	
10		92	1	8 votes in favor, we'll wait for the remaining members to attend and vote.	4	
11		92	1	So, open the roll, and call the roll please.	4	
12		2998	1	On the consent calendar, Senators Allen? Bates? Bates aye. Gaines?	0	
13		2998	1	On the consent calendar, Senators Allen? Bates? Bates aye. Gaines?	0	
14		92	1	9-0, we'll keep it on call for remaining Senators to vote.	0	
15		92	1	9-0, we'll keep it on call for remaining Senators to vote.	4	
16		92	1	Lets call the roll please.	4	
17		92	1	Lets call the roll please.	0	
18		92	1	On the consent calendar, Senators Allen? Gaines? Aye. Gaines aye.	0	
19		24	1	That's it.	0	
20		92	1	Okay that vote is 10-0, we'll remain open for the remaining member to vote, Senator Allen.	4	

Figure 4.3: One of the annotated transcripts that were used as a training set for the classifier

column on the right with integers for labels. Original idea was also to keep additional labels if the utterance contains more features from more than one category, however it was dropped later due to being more prone to subjective judgment of an annotator.

Classifiers

The classification module was technically not a part of this thesis, but it still needs some short introduction because it plays a crucial role in the pipeline of AI For Reporters. Different types of classifiers were experimented with in an attempt to achieve higher accuracy - it is important to keep the system robust and avoid false labeling results that can lead to completely wrong factual assumptions. Among those classifiers were binary ones for each label separately or multi-classification predictors for labels from 1 to 5, with different techniques such as Naiive Bayes, linear SVM, TF-IDF count vectorizer. Some of the accuracy results can be seen in the Table 4.1.

In the end, binary classification based on linear SVM approach turned out to be the most accurate among all the attempted versions. Moreover, to boost the accuracy even more, some preprocessing of the text proved to be useful. Manipulations like stop words exclusion and recognized named entities replacement with a placeholder ”person” or ”company” were performed on the training data set by the means of NLTK and SpaCy libraries.

4.2.3 Text Preprocessing

Essentially, some text preprocessing is required before the phenom extraction can be started. This preprocessing includes certain procedures:

- The utterances in the database were split generally into fragments of approximately the same length, meaning if a person had a long speech it would be still

	Binary Classifier for each section 0-4				
	Data Preprocessing + TF-IDF on Current Hearing Text				
	Class 0	Class 1	Class 2	Class 3	Class 4
Average F1 Score	0.5733	0.6943	0.6604	0.6398	0.7952
Average Accuracy	0.8393	0.8955	0.8707	0.8865	0.9269
	Binary Classifier for each section 0-4				
	Data Preprocessing + TF-IDF on Previous Hearing Text plus Current Hearing Text				
	Class 0	Class 1	Class 2	Class 3	Class 4
Average F1 Score	0.4897	0.6412	0.5489	0.6857	0.7655
Average Accuracy	0.8438	0.8786	0.8561	0.8943	0.9134

Table 4.1: Scoring results of different classifiers tested on the labeled data.

divided into several consecutive utterances by the same person. However, for the needs of this project all the consequent utterances from the same speaker has to be joined into one.

- A lookup table is build - a hash table with person identification numbers (**pid**) as keys and names and surnames as values.
- A list of experts is accumulated - all **pid**s of the people who meet certain criteria are collected in one list for further usage. The criteria was defined as following: the person must be not a legislator and has to have a speech long enough to be labeled as an expert. The threshold length of the utterance was derived from checking average lengths of the speakers in the testimony.
- All the mentions of any bill numbers are all checked and unified to one common pattern. Sometimes the Digital Democracy transcription process produces some rare spelling errors, due to what the bill names can be misspelled. Moreover, not all the speakers call the bills the same way - some prefer to say ”assembly/senate bill”, while the others will just call it ”AB/SB”. After the unification using regular expressions all the recognized bill names look like ”AB #” or ”SB #”, where # is the bill number. This allows the system to identify other bill mentions in the utterances way easier.
- An additional column is added to the DataFrame containing the word count for each utterance. Another additional column contains the SpaCy Span object of each utterance with tokenized text. This is done once for the whole hearing in the very beginning to get better performance time and don’t call the nlp pipe for tokenization of the hearing text anymore.
- In the end, based on the previously calculated word counts for each utterance, the length of the whole hearing is calculated in words to get an understanding of the scale of the transcript.
- Some discrepancies in DataFrame column names are resolved too before the main work to unify the terms and avoid KeyErrors in addressing the DataFrame by indices. All accidental Null-values that were retrieved from the database in

any cells have to be removed again for the sake of smooth work of the algorithm.
- Such preprocessed table is logged on every execution to an Excel datasheet for easier debugging - a programmer can look it up anytime and now exactly with what the program was working on this current run.

All the values and tables being calculated in this process are stored either in the DataFrame - like the word count or joined utterances - or otherwise saved in a global variable of the module for further use.

Another important process to consider before the summarization begins - the hearing has to be split into fragments determined by the paragraph classifier. Different phenoms require different segments of the text to work with, it can be either the whole hearing text or any of the five predefined paragraphs. A class `Discussion` is defined for this purpose, with fields storing six different DataFrames. The first one is the whole text, and the other five DataFrames contain only paragraphs of one type. In the process of paragraph fragmentation all the utterances within two same paragraph labels are regarded to be of the same type and added to the corresponding DataFrame. The length of each fragment in the Discussion class instance is calculated and checked for not being null - otherwise a postcondition is added about certain fragment being absent from the discussion, which is later taken into consideration by the partial order planner described in Subsection 4.2.4.

4.2.4 Facts Extraction: Phenom System

In this thesis a novel approach is introduced - a concept of ”phenom”-based fact extraction. After exploring various committee hearing transcripts, certain patterns could be spotted among the texts that provide some important or interesting information. Surely, a neural network can be trained on pre-labeled datasets to recognize such salient fragments like it was described in some of the approaches given in Section 2.4. However, the idea of what is considered ”interesting” and ”important” is a very subjective concept and may vary from person to person. The project described in this thesis is oriented on delivering summaries as a product to various end users, whose requirements may vary too. Creating a dataset and labeling it for every different need is a long and tedious process and does not seem to be a reasonable approach. Thus a need for some flexible and versatile module mechanism became evident in this project - the system must contain some easily interchangeable segments that can be included or excluded, or new ones can be added as well, like it was already stated in the system requirements in Section 3.1.

It was suggested to represent such modules as phenoms - a class that can go through the data provided to it and look for some specific facts that it can extract. Each phenom should be independent of the others unless there is a certain entailment relationship between two phenoms and one of them allows the emerging of another.

Phenom Structure

Such modules should have some common structure to unify the instantiation, storage and calls made to each of them. A decision followed up to create an abstract base class `Phenom` and make every single phenom a subclass of it, inheriting some common methods and overriding the others that have to be phenom-specific. The abstract class Phenom contains the following attributes:

- `facts` - a dictionary collecting all the facts provided by the phenom, that are later used for template filling
- `candidate_text` - a string containing a filled template sentence if it was completed correctly
- `people` - a dictionary containing all the facts on the people mentioned by the phenom
- `footnote` - a string with the background information for the facts for transparency reasons
- `completed` - a boolean value showing whether the phenom has been already executed once
- `postconditions` - a list of the postconditions generated by the phenom upon execution
- `type` - denotes a specific type of a phenom such as ”introduction” or ”summary” for further article building by themed paragraphs
- `is_pullquote` and `is_headline` - boolean values defining whether the resulting text should be handled differently in case it is not an article sentence, but a headline or a pull quote

The `Phenom` class contains several important methods that the inheriting classes use either the same way or override with its own ones. Method `check_preconditions` is by the planner to actually check whether the phenom is ready for execution and all the prerequisites are met. It returns a boolean value correspondingly to whether that's true or false. This method is inherited from the base abstract class. Method `build_phenom` is also inherited and remains the same for all the phenoms - if all the required facts were successfully gathered by the phenom, this method retrieves all the templates with corresponding identifiers and attempts to fill them in with the facts, randomly picking one afterwards. Last but not least, the method `get_facts` is abstract and overridden by each phenom differently, because each of them follow different procedures to procure the required facts. The global collection of facts is also passed to this method from the bigger scope, so that if the phenom needs some facts that already exist in the system - it can just pull it from there. This way the same procedures don't have to be repeated and the efficiency of the system is improved greatly.

Simplified Partial Order Planner

On the testing step of the phenom system, the program just had all of them hardcoded in a certain order to try out the possibilities and abilities of such a mechanism. How-

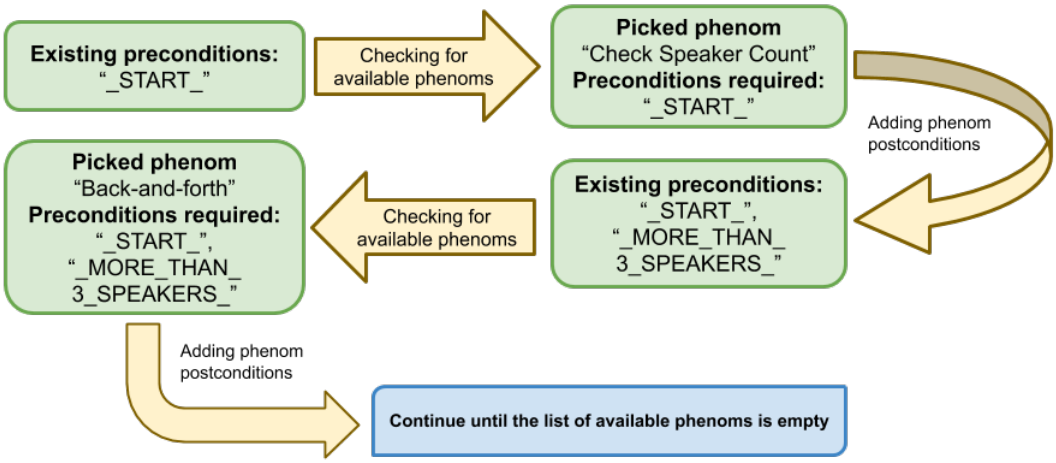

Figure 4.4: Simplified example of the postcondition-precondition planning system within AI For Reporters.

ever, as soon as phenom modules became unified, the need for a more sophisticated approach became apparent. Some inspiration was taken from such algorithms in partial order planning (POP) as STRIPS (Fikes & Nilsson, 1971) and the likes of it, with the main idea of sets of preconditions and postconditions. However, the big difference between systems with such algorithms and AI For Reporters is that normally in POP the system provides the plan beforehand, without actually executing, because the postcondition-precondition sets can be clearly calculated at any hypothetical step. On the other hand, the phenom system is built in such way, that some of them can be executed not completely and still produce some postconditions. Thus, the new set of preconditions can be only known after the finished execution of a phenom and it differs from text to text. This means, that the planning has to be combined with parallel execution and has to be reconsidered at every step. Moreover, algorithms like STRIPS rely on the end state of the plan, which is in this case unknown and cannot be specified. Furthermore, while most of the POP approaches utilizes the Principle of Least Commitment (Weld, 1994) where the goal is to accomplish the end state with the least number of steps, while in case of this thesis it is desirable to execute as many steps as possible to have a richer and bigger article containing all the information available.

Taking all this remarks into consideration, a step-by-step planning technique was devised. In the Figure 4.4 a simplified planning diagram can be seen illustrating the approach. The system starts with only starting precondition `"_START_"` and by calling the method `check_preconditions` on this precondition set for each phenom possible in the system, a list of phenoms available for execution is collected. The planner picks randomly one of them and runs the fact collection procedure. Each phenom has a list of postconditions that it can produce upon the completion of its steps, which are later

added to the set of preconditions for further choosing of next phenoms. Such a planner runs in a loop checking upon the list of available phenoms on each step and as soon as the list turns up to be empty - it's work is finished.

There are cases when some phenoms require the other ones to be executed first to be picked themselves, they require some specific postcondition generated by them as a precondition for themselves. However, it might be so that on such a step the precondition list would allow several other phenoms to be picked too. It is essential to preserve such an entailment relation between those two phenoms and ensure that the following one gets picked directly after the first one. Some addition had to be made to the planner algorithm to meet this requirement. An artificial precondition `"_HAS_PRIORITY_"` was added to the starting set of the preconditions and in the requirements of the entailment phenoms. If the planner recognizes among the available phenoms one with such precondition it is forced to pick this one first. This way some extent of order enforcement can be added to the randomness of the planner to establish the smooth flow of the article text. The steps of the plan are collected in a list in the process, which is later utilized to build the article from the sentences that each phenom has created.

4.2.5 Template-Based Sentence Generation

All the templates used by the system are stored in a shared Google spreadsheet for easier access of the non-programmer contributors. The rows of the spreadsheet contain a template identifier, template text and a list of fact identifiers present in the template. Each template is directly connected to a phenom, that mines the facts to fill this particular template. They are represented with instances of Python String Templates class, which is an extension of String class containing some variable placeholders within the text marked as "$identifier". These placeholders can be replaced by values from a dictionary under the key named the same as the identifier of a placeholder. Furthermore, this class offers two different methods for filling these template strings - `safe_substitute` which replaces all the placeholders possible and leaves the ones that have nothing to be filled with as is, and just `substitute` that raises a KeyError if any identifier appears to be missing among the keys of the mapping dictionary. The second method is the one that proved to be useful in the project for the sake of it's robustness. Ordinarily, the algorithm should not even reach the template filling step of the phenom execution if some facts are missing. However, if a template is still attempted to be filled for any reason, nothing will be produced as a result if the program does not have all the facts required.

The system supports multiple templates per phenom - firstly all the available options are being filled with facts if possible, and one of the resulting sentences is picked randomly afterwards as a candidate text. Such a variation allows the system to generate slightly different texts on each new run, producing a result that imitates more

human-written abstracts with all its language variety.

After each successful phenom execution that produced a sentence for a summary makes an update call to the output JSON structure saving the created data within the object.

4.2.6 Output Production: The JSON Collection

To make the result of the program usable and functional for the end user, a decision was made to create a JSON-structured output file, containing not only the end article, but also all the information collected over the process of running the algorithms, all the metadata required, data about the people represented in the summary, links to videos and pictures giving background to the article. The base structure of this JSON file is stored in a Google spreadsheet containing the names of the main fields and subfields, restrictions on the types of the data stored in them and other restrictions like a check on phenom names that can be added to the JSON. The main fields in this JSON structure are:

- **headline_text** - stores the headline for the article generated by the system
- **byline_text** - stores the line with author info about the article
- **date_text** - stores the date of the hearing
- **article_text** - the field for main article text storage
- **article_html** - the same article text as about, only with html markup for proper display on the html page
- **endnotes_text** - licensing line containing words like ”All rights reserved”
- **assets** - links to all the assets for the article, including images, sources, videos, etc.
- **content** - a list of substructures each containing a fact retrieved by a phenom
- **pullquotes** - all the pull quotes collected by the phenom system
- **personas** - information about all the people that are mentioned in the end summary article

This structure is read and parsed to an empty JSON structure upon the start of the execution of the program, later being updated every time some new data is mined or pulled from the database. If the new data is a phenom produced sentence - it is all added to the ”content” field, while the pull quotes are collected separately in it’s own field, and all the information about all the mentioned people is saved in ”personas” list. An example of such a JSON output file can be seen in the Listing 4.1.

4.2.7 Article assembly

After all the phenoms are finished running and the program received a complete plan, the final article assembly begins. A special method parses over the resulting JSON structure, mainly over the ”content” list, appending the sentences in the order

Listing 4.1: An example of a completed JSON output file

```
{"pullquotes": [
        {"quote_author": 109768,
        "quote_text": "Canada, Australia, Finland, South
           Korea, Czech Republic, to name a few, already
           teach their elementary and high school students to
            be media literate.",
        "quote_note_url": null,
        "quote_caption_full": null,
        "quote_citation": "pull_quote_extractor",
        "quote_author_affiliation": "Beth Thorton, a member
           of the Center for Media Literacy",
        "quote_note": "The pull quote is retrieved by the
           Pull Quote Extractor Module."}],
"article_text": "In California on Wednesday...",
"personas": [
        {"pid": 113,
        "info": "Patrick O'Donnell, Democratic Assembly
           member representing district 70",
        "note": "chairperson",
        "last": "O'Donnell",
        "first": "Patrick"}, ... ],
"headline_text": "Headline for the bill discussion",
"endnotes_text": "All Rights Reserved (c). AI4Reporters,
   2020.",
"date_text": "Wednesday, July 12, 2017",
"byline_text": "AI4Reporters",
"content": [
        {"text": "In California on Wednesday, July 12, 2017,
           Assembly Standing Committee on Education met and
           discussed the bill SB135.",
        "phenom": "intro",
        "note": "Extracted from Digital Democracy Records",
        "citation": null},
        {"text": "The official title of the bill SB135 is:
           An act to add Section 51206.3 to the Education
           Code, relating to pupil instruction. .",
        "phenom": "bill_name",
        "note": "Extracted from Digital Democracy records",
        "citation": null}, ... ]
}
```

according to the devised plan. If a footnote is found to a sentence, a footnote symbol has to be appended to the end of the sentence and the footnote text to the end of the article. If any pull quotes are found among the data, they are inserted at some point inside the article, formatted properly with tabulations and quotation marks to stand out from the text.

As it was mentioned before, there can be some variations in the planning due to the random picking of the phenoms. Moreover, multiple template availability per phenom also adds up to this diversity, meaning that the end product might vary on different runs of the program. This will help to make the articles sound more natural and less robotic, especially if the tool will be used repeatedly in one source for various transcripts.

4.2.8 Reverse Anaphora Resolution Problem

In human speech or even written text it is absolutely natural to introduce an object or a person for the first time with a full name maybe even with some titles or qualities, however in later mentions of the same entity refer to it with a shortened name or even a pronoun. One of the goals of this thesis is to attempt to mimic human-written abstracts, thus such phenomena has to be taken into consideration. A computer program producing sentences that may mention the same entities repeatedly should have some mechanism to reproduce this phenomenon, some technique to keep track of what has been already mentioned and what is being introduced for the first time.

Within this thesis project this problem arises regarding the names of the legislators and public mentioned in the generated sentences. An approach was suggested, since every person that has ever taken part in any hearing is documented in the Digital Democracy database with their own ID and affiliation information if any is available. This way, the templates can be filled not directly with the names, but with personal IDs, keeping the data stored behind this ID in some collection within the JSON. Afterwards, a post-processing step can be applied to an already assembled article, counting the mentions and replacing them with either full names with affiliations or shorter versions of names. This way the problem becomes some sort of reverse anaphora resolution problem (Mitkov, 2014), however in this case the algorithm has to populate different references for the same entity instead of finding the different ones and bringing them to one form.

To begin with, all the placeholders for people in the templates are preceded with double underscores to make it easier to find the personal IDs in the text later. The algorithm within AI For Reporters utilizes regular expressions and with the help of ”re” library searches through the text for all the IDs, replacing them with names one by one. An empty dictionary is created beforehand to keep track of repetitions. Whenever an ID is found, it is checked over the dictionary, and if it is not present, then a full name

with titles and affiliations is placed in the text instead of the ID and it is saved in the dictionary. If the ID already exists in the dictionary, the full name is reduced to just the last name or the last name with title. For example, a person with the personal ID 113 when mentioned for the first time will be referred as ”Patrick O’Donnell, Democratic Assembly member representing district 70”. For the second and next times it will be just ”Assembly Member O’Donnell” or even simply ”O’Donnell”. Such a method proved to be efficiently coping with this issue, bringing more flow to the texts.

4.3 Summary

In this Chapter the actual design of the project corresponding to the needs and requirements described previously in Chapter 3 is first described in detail and then carried out and the implementation process is explained with technical features and examples. The structure has been adapted slightly several times during the development process with some components being scrapped, improved or changed, increasing the execution time, flexibility of the system and it’s capability to expand to further needs and ideas in the future. Such changes and decisions will be discussed in more details in Chapter **??**.

5 Evaluation

In the following Chapter the approach to evaluation of the project is discussed, contemplating on the ways to assess the quality of the summarization. Two main research questions are established for the user study - the evaluation of the factual quality of the summary, the coverage and correctness, as well as the grammaticality, coherence and the flow of the text.

The user study aims to check the following hypotheses:

- Can effective, original natural language headlines referring to content of a particular hearing or bill discussion be generated automatically?
- Can legislative proceedings be effectively summarized using fully automated abstractive methods, given the full proceedings and associated metadata?
- Will automated summaries be sufficiently informative and interesting to readers akin to human generated ones?
- Can automated reference generation allow readers to trace every claim made within the summaries, to a primary source fact or video documentation?

Remark: due to the current pandemic situations of 2020 in the US, the in-person user study had to be canceled, so by the time of writing this report no actual evaluation data can yet be obtained. Thus this Chapter is going to concentrate mainly on the structure of the user study and the approach taken for the testing of the implementation. Evaluation results will be included and analyzed in the full version of this master thesis.

5.1 Experiment

Many of the summarization systems require various metrics to evaluate the quality of produced texts, such as coherence, content, grammaticality, readability (Mani, 2001), etc. At the beginning of the research in the field of text summarization such evaluation tests had to be performed manually with the help of human experts, which is costly and time-consuming. Realizing those drawbacks, the researchers came up with various automated systems with build in metrics for summary assessment. Saggion, Radev, Teufel, and Lam (2002) suggested three techniques for content-based evaluation: cosine similarity, unit overlap and longest common subsequence. Papineni, Roukos, Ward, and Zhu (2002) offered a application of automated evaluation methods called BLEU (stands for Bilingual Evaluation Understudy) while Lin (2004) proposed later the system called

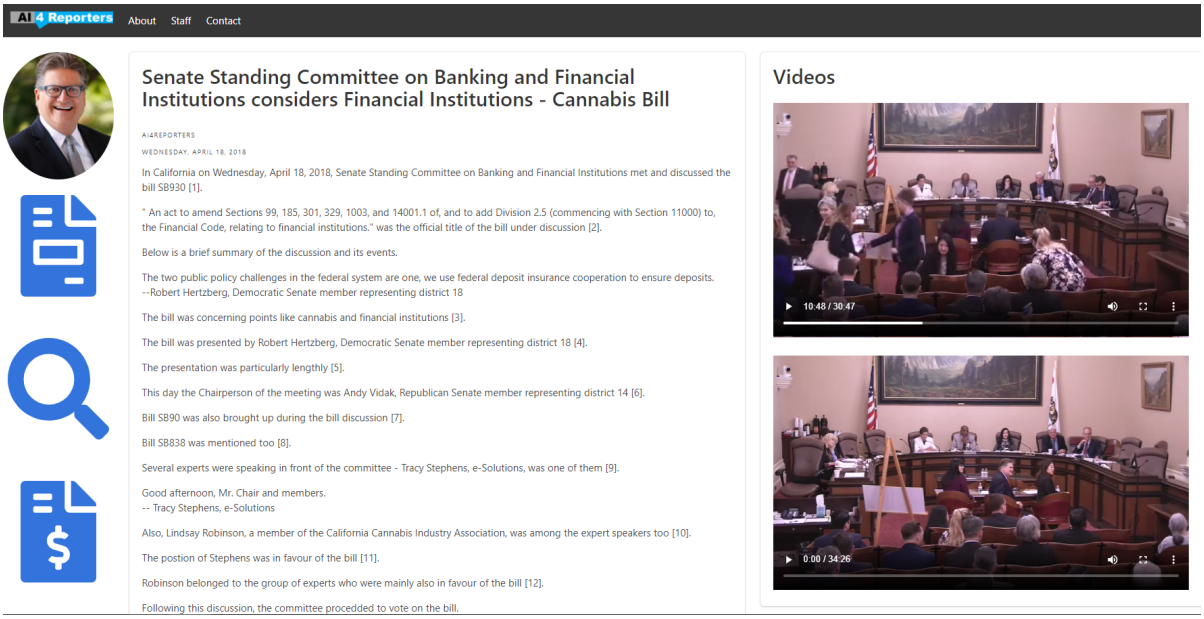

Figure 5.1: AI For Reporters webpage.

ROUGE (stands for Recall-Oriented Understudy for Gisting Evaluation). However, all those systems rely on a comparison of the summary to some gold-standard abstract, usually human written.

Unfortunately, in case of AI For Reporters, there is no human texts to compare to, so the user study has to resort to old-fashion ways of evaluation. On the other hand, human-conducted tests are easier and cheaper to crowdsource than a couple of decades ago. Moreover, such services also suddenly brought a greater variety to the demographics of survey respondents, which in university-based works were mainly found among student population (Samuel, 2018). Amazon Mechanical Turk ("Amazon Mechanical Turk," 2020) is an example of such a crowdsourcing mechanism and it was decided to be used for a remote user study for this project.

The turkers will be introduced to the purposes of the system, and will be required to watch several recordings of the committee hearings and answer some easy questions about the contents to make sure that the turkers actually have watched the videos. Afterwards they will be redirected to a webpage representation of the AI For Reporters output (see Figure 5.1), where the summary text is rendered with all the footnotes and assets like videos, links, images available. The turkers will have to read through the summary article and answer more questions afterwards in a questionnaire. Likert scale (Joshi, Kale, Chandel, & Pal, 2015) is used in most of the questions that are, as was already mentioned before, split in two categories of assessment - summary quality and article quality evaluation. The questionnaire attempts to get feedback from the respondents regarding the present and the missing important facts in the summary with the comparison to the recording itself. In these questions a text field is available

to provide more informative response that can be later used for future work definition and corrections. The second category of questions presents the turkers with statements that they can indicate their extent of agreement or disagreement with. Such questions inquire about the flow of the text, how smoothly it reads, if there are any grammatical or spelling mistakes, if the article approaches the quality of the human-written abstracts or not.

Analyzing the responds to these questions will help to understand the efficiency of the system and will show whether the phenom extraction approach leads to a successful combination of extractive and abstractive summarization techniques.

5.2 Summary

In this chapter an approach to the evaluation of such a summarization system was discussed, contemplating on the pros and cons of it and justifying the choice. The evaluation procedure is described afterwards, explaining how certain questions in the user study help to prove the research hypotheses defined in this work.

6 Conclusion and Future Work

This chapter draws conclusion regarding the accomplished tasks. It also brings up some possible further improvements and adaptations of the system that might boost up the effectiveness of the summarization tool and help to create better summaries closer to the human-written abstracts.

6.1 Conclusion

Within the project an automated system generating summary articles based on the transcripts and data on legislation proceedings was successfully created. This work achieved several lesser goals as well:

- A paragraph classification mechanism for committee hearing transcripts was devised
- The idea of the phenom extraction based approach was carried out and implemented in a fully functional algorithm
- A planning technique was applied to the set of the phenoms available for a dynamic creation of a summary article
- A step towards news transparency was taken by adding sources and footnotes to any fact brought up in the generated sentences

6.2 Future Work

Several improvements can still be added to the system afterwards to enhance the results. First, the paragraph classifier efficiency is satisfactory for the task but still not perfect - it is possible that expanding the training dataset or changing the approach to text preprocessing can yield better results and boost the accuracy of the classifier. Second, some other techniques of template creation other than using human-written ones could be interesting to test, adapting some of the approaches to template generation from the works discussed in Section 2.4. Third, even more sophisticated article planner may also help in creating the article with better flow and fluency.

In general, since the system is expandable so easily by just adding new phenoms, it could be a big step ahead to get connected with some colleagues from journalism or political studies and collaborate to create more phenoms involving some other patterns from the transcripts that were missed out on in this work. Furthermore, some more

complicated and sophisticated phenoms can be added, for example, analyzing tones and sentiments, or keeping track of the discourse by the means of attentions networks or other methods.

Involving the recordings videos or even fragments of it could be another possible future improvement. The idea of combining text footnotes with sources to the facts and actual videos with the timestamps for those sources was discussed as a future addition during the development process. Such an improvement can help to add up even more to the transparency of the news article.

Lastly, involving some video extractive summarization could be an interesting theory to test out. A hypothesis whether combining video excerpts with textual summary information will provide better experience for the user and yield more informative summarization remains a question for future research on the topic.

Bibliography

Amazon Mechanical Turk. (2020). Retrieved August 7, 2020, from https://www.mturk.com/

Anderson, C. W. (2013). Towards a sociology of computational and algorithmic journalism. *New media & society*, *15*(7), 1005–1021.

Association, G. W. (n.d.).

Austrian Marshall Plan Foundation. (2020). Retrieved August 7, 2020, from https://www.marshallplan.at/

Babar, S., & Patil, P. D. (2015). Improving performance of text summarization. *Procedia Computer Science*, *46*, 354–363.

Banerjee, S., Mitra, P., & Sugiyama, K. (2015). Abstractive meeting summarization using dependency graph fusion. In *Proceedings of the 24th international conference on world wide web* (pp. 5–6).

Barzilay, R., & Elhadad, M. (1999). Using lexical chains for text summarization. *Advances in automatic text summarization*, 111–121.

Baxendale, P. B. (1958). Machine-made index for technical literature—an experiment. *IBM Journal of Research and Development*, *2*(4), 354–361.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python* (1st). O'Reilly Media, Inc.

Blakeslee, S., Dekhtyar, A., Khosmood, F., Kurfess, F., Poschman, H., Prinzivalli, G., … Durst, S. (2015). Digital democracy project: Making government more transparent one video at a time, Sydney, Australia: Digital Humanities Conference.

Boehner, K., & DiSalvo, C. (2016). Data, design and civics: An exploratory study of civic tech. In *Proceedings of the 2016 chi conference on human factors in computing systems* (pp. 2970–2981).

Boguraev, B., & Kennedy, C. (1999). Salience-based content characterisation of text documents. *Advances in automatic text summarization*, 99–110.

Brewer, M. B. (1999). The psychology of prejudice: Ingroup love and outgroup hate? *Journal of social issues*, *55*(3), 429–444.

Budhwar, A., Kuboi, T., Dekhtyar, A., & Khosmood, F. (2018). Predicting the vote using legislative speech. In *Proceedings of the 19th annual international conference on digital government research: Governance in the data age*. dg.o '18. doi:10.1145/3209281.3209374

Bui, T., Frampton, M., Dowding, J., & Peters, S. (2009). Extracting decisions from multi-party dialogue using directed graphical models and semantic similarity. In *Proceedings of the sigdial 2009 conference* (pp. 235–243).

California Legislative Information. (2020). Retrieved August 7, 2020, from https://leginfo.legislature.ca.gov

Capilla, R., Babar, M. A., & Pastor, O. (2012). Quality requirements engineering for systems and software architecting: Methods, approaches, and tools. *Requirements Engineering*, *17*(4), 255–258.

Carlson, L., Marcu, D., & Okurowski, M. E. (2003). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue* (pp. 85–112). Springer.

Chen, S.-C., Chang, J.-S., Wang, J.-N., & Su, K.-Y. (1991). Archtran: A corpus-based statistics-oriented english-chinese machine translation system. In *Proceedings of machine translation summit iii* (pp. 33–40).

Chomsky, N. (1957). *Syntactic structures*. Mouton.

Christensen, H., Gotoh, Y., Kolluru, B., & Renals, S. (2003). Are extractive text summarisation techniques portable to broadcast news? In *2003 ieee workshop on automatic speech recognition and understanding (ieee cat. no. 03ex721)* (pp. 489–494). IEEE.

Cohen, S., Hamilton, J. T., & Turner, F. (2011). Computational journalism. *Communications of the ACM*, *54*(10), 66–71.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Crawford, C. (2006). *Attack the messenger: How politicians turn you against the media*. Rowman & Littlefield.

Dawes, S. S., & Helbig, N. (2010). Information strategies for open government: Challenges and prospects for deriving public value from government transparency. In M. A. Wimmer, J.-L. Chappelet, M. Janssen, & H. J. Scholl (Eds.), *Electronic government* (pp. 50–60). Berlin, Heidelberg: Springer Berlin Heidelberg.

Dawson, R. (2010). The rise of robot journalists. Retrieved August 7, 2020, from https://rossdawson.com/blog/the_rise_of_rob/

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the ACM (JACM)*, *16*(2), 264–285.

Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, *22*, 457–479.

Eur-LEX: Access To European Union Law. (2020). Retrieved August 7, 2020, from https://eur-lex.europa.eu/homepage.html

Fernández, R., Frampton, M., Ehlen, P., Purver, M., & Peters, S. (2008). Modelling and detecting decisions in multi-party dialogue. In *Proceedings of the 9th sigdial workshop on discourse and dialogue* (pp. 156–163).

Fikes, R. E., & Nilsson, N. J. (1971). Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, *2*(3-4), 189–208.

Filippova, K. (2010). Multi-sentence compression: Finding shortest paths in word graphs. In *Proceedings of the 23rd international conference on computational linguistics (coling 2010)* (pp. 322–330).

Fischer-Hwang, I., Grosz, D., Hu, X. E., Karthik, A., & Yang, V. (2020). Disarming loaded words: Addressing gender bias in political reporting. Computation+ Journalism'20 Conference, Boston, MA.

Franklin, B., & Carlson, M. (2010). *Journalists, sources, and credibility: New perspectives*. Routledge.

Galley, M., McKeown, K., Fosler-Lussier, E., & Jing, H. (2003). Discourse segmentation of multi-party conversation. In *Proceedings of the 41st annual meeting of the association for computational linguistics* (pp. 562–569).

Gambhir, M., & Gupta, V. [Vishal]. (2017). Recent automatic text summarization techniques: A survey. *Artificial Intelligence Review*, *47*(1), 1–66.

Ganesh, P., & Dingliwal, S. (2019). Abstractive summarization of spoken and written conversation. *arXiv preprint arXiv:1902.01615*.

Goldberg, E., Driedger, N., & Kittredge, R. I. (1994). Using natural-language processing to produce weather forecasts. *IEEE Expert: Intelligent Systems and Their Applications*, *9*(2), 45–53. doi:10.1109/64.294135

Gottfried, J. (2020, February). Americans' news fatigue isn't going away – about two-thirds still feel worn out. Retrieved August 7, 2020, from https://www.pewresearch.org/fact-tank/2020/02/26/almost-seven-in-ten-americans-have-news-fatigue-more-among-republicans/

Graefe, A., Haim, M., Haarmann, B., & Brosius, H.-B. (2018). Readers' perception of computer-generated news: Credibility, expertise, and readability. *Journalism*, *19*(5), 595–610. doi:10.1177/1464884916641269. eprint: https://doi.org/10.1177/1464884916641269

Grieco, E. (2020, April). U.s. newspapers have shed half of their newsroom employees since 2008. Retrieved August 7, 2020, from https://www.pewresearch.org/fact-tank/2020/04/20/u-s-newsroom-employment-has-dropped-by-a-quarter-since-2008/

Guerra, A. (2001). T. rowe price to hone in on voice systems. *Wall Street & Technology*, *19*(3), 11–11.

Gupta, V. [Vikrant], Chauhan, P., Garg, S., Borude, A., & Krishnan, S. (2012). An statistical tool for multi-document summarization. *International Journal of Scientific and Research Publications*, *2*(5).

Hariharan, S., Ramkumar, T., & Srinivasan, R. (2013). Enhanced graph based approach for multi document summarization. *Int. Arab J. Inf. Technol.*, *10*(4), 334–341.

Hayes-Roth, F. (1985). Rule-based systems. *Commun. ACM*, *28*(9), 921–932. doi:10.1145/4284.4286

Henke, J., Leissner, L., & Möhring, W. (2020). How can journalists promote news credibility? effects of evidences on trust and credibility. *Journalism Practice*, *14*(3), 299–318.

Hirschberg, J., & Manning, C. D. [Christopher D]. (2015). Advances in natural language processing. *Science*, *349*(6245), 261–266.

Honnibal, M., & Montani, I. (2017). *spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing*. To appear.

Huang, L., He, Y., Wei, F., & Li, W. (2010). Modeling document summarization as multi-objective optimization. In *2010 third international symposium on intelligent information technology and security informatics* (pp. 382–386). IEEE.

Jing, H. (2000). Sentence reduction for automatic text summarization. In *Sixth applied natural language processing conference* (pp. 310–315).

Jing, H., & McKeown, K. (2000). Cut and paste based text summarization. In *1st meeting of the north american chapter of the association for computational linguistics*.

Joshi, A., Kale, S., Chandel, S., & Pal, D. K. (2015). Likert scale: Explored and explained. *Current Journal of Applied Science and Technology*, 396–403.

Karen Callaghan, F. S. (2001). Assessing the democratic debate: How the news media frame elite policy discourse. *Political Communication*, *18*(2), 183–213. doi:10.1080/105846001750322970. eprint: https://doi.org/10.1080/105846001750322970

Kazimierczak, J. (1990). An approach to natural language processing in the rule-based expert system. In *Proceedings of the 1990 acm annual conference on cooperation* (pp. 215–222). CSC '90. doi:10.1145/100348.100381

Kincaid, J. P., Fishburne Jr, R. P., Rogers, R. L., & Chissom, B. S. (1975). *Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel*. Naval Technical Training Command Millington TN Research Branch.

Kupiec, J., Pedersen, J., & Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th annual international acm sigir conference on research and development in information retrieval* (pp. 68–73).

Kurtz, K. T. (1997). *Legislatures and citizens*.

Lahav, H., & Reich, Z. (2011). Authors and poets write the news: A case study of a radical journalistic experiment. *Journalism Studies*, *12*(5), 624–641.

Lathrop, D., & Ruma, L. (2010). *Open government: Collaboration, transparency, and participation in practice*. " O'Reilly Media, Inc."

Latner, M., Dekhtyar, A. M., Khosmood, F., Angelini, N., & Voorhees, A. (2017). Measuring legislative behavior: An exploration of digitaldemocracy. org. *California Journal of Politics and Policy*, *9*(3).

Lee, E.-J., & Kim, Y. W. (2016). Effects of infographics on news elaboration, acquisition, and evaluation: Prior knowledge and issue involvement as moderators. *New Media & Society*, *18*(8), 1579–1598. doi:10.1177/1461444814567982. eprint: https://doi.org/10.1177/1461444814567982

Leppänen, L., Tuulonen, H., Sirén-Heikel, S., et al. (2020). Automated journalism as a source of and a diagnostic device for bias in reporting. *Media and Communication*.

Levy, S. (2018). Can an algorithm write a better news story than a human reporter? Conde Nast. Retrieved August 7, 2020, from https://www.wired.com/2012/04/can-an-algorithm-write-a-better-news-story-than-a-human-reporter/

Li, J., Luong, M.-T., & Jurafsky, D. (2015). A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out* (pp. 74–81).

Linden, T. C.-G. et al. (2017). Algorithms for journalism: The future of news work. *The journal of media innovations*.

Liu, F., & Liu, Y. (2009). From extractive to abstractive meeting summaries: Can it be done by sentence compression? In *Proceedings of the acl-ijcnlp 2009 conference short papers* (pp. 261–264).

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM J. Res. Dev.*, *2*(2), 159–165. doi:10.1147/rd.22.0159

Mani, I. (1999). *Advances in automatic text summarization* (M. T. Maybury, Ed.). Cambridge, MA, USA: MIT Press.

Mani, I. (2001). *Automatic summarization*. John Benjamins Publishing.

Manjoo, F. (2013, July). You won't finish this article. why people online don't read to the end. Retrieved August 7, 2020, from https://slate.com/technology/2013/06/how-people-read-online-why-you-wont-finish-this-article.html

Manning, C. D. [Christopher D.], & Schütze, H. (1999). *Foundations of statistical natural language processing*. Cambridge, MA, USA: MIT Press.

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, *19*(2), 313–330. Retrieved from https://www.aclweb.org/anthology/J93-2004

Matsa, K. E., & Boyles, J. L. (2014, July). America's shifting statehouse press. Retrieved August 7, 2020, from https://www.journalism.org/2014/07/10/americas-shifting-statehouse-press/

Mc Laughlin, G. H. (1969). Smog grading-a new readability formula. *Journal of reading*, *12*(8), 639–646.

McDermott, P. (2010). Building open government. *Government Information Quarterly*, *27*(4), 401–413.

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). doi:10.25080/Majora-92bf1922-00a

Mehdad, Y., Carenini, G., & Ng, R. (2014). Abstractive summarization of spoken and written conversations based on phrasal queries. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1220–1230).

Mehdad, Y., Carenini, G., Tompa, F., & Ng, R. (2013). Abstractive meeting summarization with entailment and fusion. In *Proceedings of the 14th european workshop on natural language generation* (pp. 136–146).

Meyer, R. (2016). How many stories do newspapers publish per day? Retrieved from https://www.theatlantic.com/technology/archive/2016/05/how-many-stories-do-newspapers-publish-per-day/483845/

Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 404–411).

Mitkov, R. (2014). *Anaphora resolution*. Routledge.

Murray, G. (2015). Abstractive meeting summarization as a markov decision process. In *Canadian conference on artificial intelligence* (pp. 212–219). Springer.

Murray, G., & Carenini, G. (2008). Summarizing spoken and written conversations. In *Proceedings of the 2008 conference on empirical methods in natural language processing* (pp. 773–782).

Murray, G., Carenini, G., & Ng, R. (2010). Generating and validating abstracts of meeting conversations: A user study. In *Proceedings of the 6th international natural language generation conference* (pp. 105–113). INLG '10. Trim, Co. Meath, Ireland: Association for Computational Linguistics.

Nadkarni, P., Ohno-Machado, L., & Chapman, W. (2011). Natural language processing: An introduction. *Journal of the American Medical Informatics Association : JAMIA*, *18*, 544–51. doi:10.1136/amiajnl-2011-000464

Nenkova, A., & McKeown, K. (2011). Automatic summarization. *Foundations and Trends® in Information Retrieval*, *5*(2–3), 103–233. doi:10.1561/1500000015

Nesterenko, L. (2016). Building a system for stock news generation in russian. In *Proceedings of the 2nd international workshop on natural language generation and the semantic web (webnlg 2016)* (pp. 37–40).

Niven, D. (2002). *Tilt?: The search for media bias*. Greenwood Publishing Group.

Oliphant, T. E. (2006). *A guide to numpy*. USA: Trelgol Publishing.

Oya, T., Mehdad, Y., Carenini, G., & Ng, R. (2014). A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th international natural language generation conference (inlg)* (pp. 45–53).

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the association for computational linguistics* (pp. 311–318).

Parks, B. (2014). Basic news writing.

Parliamentary Office of Science and Technology. (2009). E-democracy, post pn321. Retrieved August 7, 2020, from https://post.parliament.uk/research-briefings/post-pn-321/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Peiser, J. (2019). The rise of the robot reporter. Retrieved August 7, 2020, from https://www.nytimes.com/2019/02/05/business/media/artificial-intelligence-journalism-robots.html

PEW Research Center. (2009, March). Nearly as many americans prefer to get their local news online as prefer the tv set. Retrieved August 7, 2020, from https://www.journalism.org/2019/03/26/nearly-as-many-americans-prefer-to-get-their-local-news-online-as-prefer-the-tv-set/

Pöttker, H. (2003). News and its communicative quality: The inverted pyramid—when and why did it appear? *Journalism Studies*, *4*(4), 501–511.

Reshamwala, A., Mishra, D., & Pawar, P. (2013). Review on natural language processing. *IRACST Engineering Science and Technology: An International Journal (ESTIJ)*, *3*(1), 113–116.

Riedhammer, K., Favre, B., & Hakkani-Tür, D. (2010). Long story short–global unsupervised models for keyphrase based meeting summarization. *Speech Communication*, *52*(10), 801–815.

Ruprechter, T., Khosmood, F., Kuboi, T., Dekhtyar, A., & Gütl, C. (2018). Gaining efficiency in human assisted transcription and speech annotation in legislative proceedings. In *Proceedings of the 19th annual international conference on digital government research: Governance in the data age* (pp. 1–2).

Saggion, H., Radev, D., Teufel, S., & Lam, W. (2002). Meta-evaluation of summaries in a cross-lingual environment using content-based metrics. In *Coling 2002: The 19th international conference on computational linguistics*.

Salton, G. (1989). Automatic text processing: The transformation, analysis, and retrieval of. *Reading: Addison-Wesley*, *169*.

Samuel, A. (2018). Amazon's Mechanical Turk has Reinvented Research. Retrieved August 7, 2020, from https://daily.jstor.org/amazons-mechanical-turk-has-reinvented-research/

Schonfield, E. (2010, November). Automated news comes to sports coverage via statsheet. Retrieved August 7, 2020, from https://techcrunch.com/2010/11/12/automated-news-sports-statsheet/

See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Senter, R., & Smith, E. A. (1967). *Automated readability index*. CINCINNATI UNIV OH.

Shang, G., Ding, W., Zhang, Z., Tixier, A. J.-P., Meladianos, P., Vazirgiannis, M., & Lorré, J.-P. (2018). Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. *arXiv preprint arXiv:1805.05271*.

Singla, K., Stepanov, E., Bayer, A. O., Carenini, G., & Riccardi, G. (2017). Automatic community creation for abstractive spoken conversations summarization. In *Proceedings of the workshop on new frontiers in summarization* (pp. 43–47).

Stone, M., Stojnic, U., & Lepore, E. (2013). Situated utterances and discourse relations. In *Proceedings of the 10th international conference on computational semantics (iwcs 2013)–short papers* (pp. 390–396).

Su, K.-Y., Chiang, T.-H., & Chang, J.-S. (1996). An overview of corpus-based statistics-oriented (cbso) techniques for natural language processing. In *International journal of computational linguistics & chinese language processing, volume 1, number 1, august 1996* (pp. 101–158).

Sunil, R., Jayan, V., & Bhadran, V. K. (2012). Preprocessors in nlp applications: In the context of english to malayalam machine translation. In *2012 annual ieee india conference (indicon)* (pp. 221–226).

The Pandas Development Team. (2020). Pandas-dev/pandas: Pandas (Version latest). doi:10.5281/zenodo.3509134

Tu, Z., Lu, Z., Liu, Y., Liu, X., & Li, H. (2016). Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*.

Turner, F., & Hamilton, J. T. (2009). Accountability through algorithm: Developing the field of computational journalism. *Online at: http://dewitt. sanford. duke. edu/images/uploads/About_3_Research_B_cj_1_finalreport. pdf. Accessed November*, *12*, 2013.

UK Legislation Portal. (2020). Retrieved August 7, 2020, from https://www.legislation.gov.uk/

University, P. (2010). About wordnet. Retrieved August 7, 2020, from https://wordnet.princeton.edu/

Van Dalen, A. (2012). The algorithms behind the headlines: How machine-written news redefines the core skills of human journalists. *Journalism practice*, *6*(5-6), 648–658.

Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley, CA: CreateSpace.

Wang, L., & Cardie, C. (2013). Domain-independent abstract generation for focused meeting summarization. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 1395–1405).

Weizenbaum, J. (1966). Eliza: A computer program for the study of natural language communication between man and machine. *Commun. ACM*, *9*(1), 36–45. doi:10.1145/365153.365168

Weld, D. S. (1994). An introduction to least commitment planning. *AI magazine*, *15*(4), 27–27.

Winograd, T. (2004). Procedures as a representation for data in a computer program for understanding natural language.

Xie, S., Liu, Y., & Lin, H. (2008). Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *2008 ieee spoken language technology workshop* (pp. 157–160). IEEE.

Yao, J.-g., Wan, X., & Xiao, J. (2017). Recent advances in document summarization. *Knowledge and Information Systems*, *53*(2), 297–336. doi:10.1007/s10115-017-1042-4

Yong, S., Abidin, A. I., & Chen, Y. (2006). A neural-based text summarization system. *WIT Transactions on Information and Communication Technologies*, *37*.

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, *13*(3), 55–75.

Yu, R. (2014, July). How robots will write earnings stories for the ap. Retrieved August 7, 2020, from https://eu.usatoday.com/story/money/business/2014/06/30/ap-automated-stories/11799077

Zhang, J., Sun, L., & Zhou, Q. (2005). A cue-based hub-authority approach for multi-document text summarization. In *2005 international conference on natural language processing and knowledge engineering* (pp. 642–645). IEEE.

Zhao, Z., Pan, H., Fan, C., Liu, Y., Li, L., Yang, M., & Cai, D. (2019). Abstractive meeting summarization via hierarchical adaptive segmental network learning. In *The world wide web conference* (pp. 3455–3461).

Zhuge, H. (2015). Dimensionality on summarization. *CoRR*, *abs/1507.00209*. arXiv: 1507.00209. Retrieved from http://arxiv.org/abs/1507.00209