NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PEDRO SANTOS GALVÃO DE ALMEIDA

Bachelor in Computer Science and Engineering

TURN-BASED TEMPORAL MEDIA WEB VISUALIZATION AND QUERYING FOR NEWS IMAGES

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
July, 2022

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TURN-BASED TEMPORAL MEDIA WEB VISUALIZATION AND QUERYING FOR NEWS IMAGES

PEDRO SANTOS GALVÃO DE ALMEIDA

Bachelor in Computer Science and Engineering

Adviser: Rui Nóbrega
Assistant Professor, NOVA School of Science and Technology

Co-adviser: David Semedo
Assistant Professor, NOVA School of Science and Technology

Examination Committee

Chair: João Carlos Gomes Moura Pires
Associate Professor, NOVA School of Science and Technology

Rapporteur: Daniel Gonçalves
Instituto Superior Técnico, ULisboa, Associate Professor

Adviser: Rui Nóbrega
Assistant Professor, NOVA School of Science and Technology

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
July, 2022

Turn-based Temporal Media Web Visualization and Querying for News Images

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.9.6) [52].

Acknowledgements

I would like to start by thanking my supervisors, Professor Rui Nóbrega and Professor David Semedo for all the support, help with the writing, and very close guidance. I am also very grateful for the fact that the professors were always very correct with me and patient even when perhaps the professors had reasons to be frustrated with me, which I am sure they were quite a few.

I thank the Nova School of Science and Technology | FCT NOVA, for providing me with 5 years of high quality education. I am also absolutely grateful for the funding given to my thesis project through a research grant.

I would like to thank all the volunteers who participated in the user tests of my project and who allowed me to obtain results, thus giving more value to the thesis.

I thank all my friends, family, and all the people who showed interest and concern throughout the preparation and implementation of the thesis, even though it was quite annoying to be asked "when are you going to finish your thesis" almost every day.

Lastly, I would like to leave a special thanks to my college colleague and long-time friend João Vilar, for all the help he gave me throughout the duration of my thesis, even when it was nowhere near his obligation.

Once again, thank you all very much.

Abstract

Content creators and journalists benefit from having related content in their developing news. With the increasing amount of digital news available on the internet, content news creators have to explore through vast amounts of news collections in order to acquire related content to insert in news pieces. Regularly, journalists and content creators only have a partial idea of what they want to search and, many times, get overburden with the amount of information available. These problems make the task of exploring and finding relevant related content difficult.

The goal of this dissertation was to create a media oriented web application interface to facilitate the introduction and exploration of related content in a news piece, with a focus on exploring large online news collections. This interface supports a turn based querying of the system and communicates with an Embeddings System that provides recommendations based on an initial input (query) using the online New York Times feed as example of data input. In order to achieve this goal, a Data Visualization System was created providing several types of interactive visualizations, such as visual graph visualization, time analysis, embeddings visualization and other type of visualizations. Through the set of visualizations and interactions provided by the system, users manage to do a turn based exploration with each turn being a more concrete and in-depth analysis of the user's search query. Additionally, the implemented system was tested with real users in order to understand the efficacy of the several type of visualizations. In general users were able to correctly interact with the visualizations and complete the set of tasks presented to them during the testing of the system.

Keywords: Data visualization, Data interaction, HCI (human-computer interaction), Data journalism.

Resumo

Os criadores de conteúdos beneficiam de ter conteúdos relacionados nas suas notícias em desenvolvimento. Com a quantidade crescente de notícias digitais disponíveis na Internet, os criadores de notícias de conteúdo têm de explorar através de grandes quantidades de colecções de notícias, a fim de adquirirem conteúdos relacionados a inserir em peças de notícias. Regularmente, jornalistas e criadores de conteúdos têm apenas uma ideia do que querem pesquisar e, muitas vezes, ficam sobrecarregados com a quantidade de informação disponível. Estes problemas dificultam a tarefa de explorar e encontrar conteúdos relacionados relevantes.

O objectivo desta dissertação passou pela criação de uma interface de aplicação web orientada para os media para facilitar a introdução e exploração de conteúdos relacionados numa peça noticiosa, com um foco na exploração de grandes colecções de notícias em linha. Esta interface suporta uma consulta do sistema por turnos e comunica com um Sistema Embeddings que fornece recomendações baseadas numa pesquisa inicial (consulta) utilizando o feed online do New York Times como exemplo de entrada de dados. Para atingir este objectivo, foi criado um Sistema de Visualização de Dados para fornecer vários tipos de visualizações interactivas, tais como visualização de grafos, análise temporal, visualização de embeddings e outro tipo de visualizações. Através do conjunto de visualizações e interacções fornecidas pelo sistema, os utilizadores conseguem fazer uma exploração baseada em turnos, sendo cada turno uma análise mais concreta e aprofundada da consulta de pesquisa do utilizador. Além disso, o sistema implementado foi testado com utilizadores a fim de compreender a eficácia dos vários tipos de visualizações. Em geral, os utilizadores puderam interagir correctamente com as visualizações e completar o conjunto de tarefas que lhes foram apresentadas durante os testes do sistema.

Palavras-chave: Visualização de dados, Interacção de dados, IHM (interacção homem-máquina), Jornalismo de dados.

Contents

List of Figures

List of Tables

1

Introduction

The consumption of news articles is getting increasingly more digital. Every day, large amounts of news articles are released on the internet, whether through television news, newspapers and other types of news aggregators. Online news are an important information source because it informs the masses and can make an impact in various types of domains such as business, politics, science and many others. According to *The Atlantic* [1], in 2016, the newspapers *The Washington Post* and *The New York Times* releases on average 500 stories and 230 videos respectively. This has led to the creation of large document news collections which gathers news of every type of topic and throughout the time. This type of collections can be very beneficial to journalists and content creators for them to gather and search information. But finding the desired information in large collections of data can be similar to finding a needle in a haystack.

Data visualization techniques and turn-based sessions have previously played a role in finding valuable information in large collections, as visualizations can often take advantage of the human cognitive abilities, helping breaking seemingly complex problems into more understandable ones.

1.1 Motivation and Problem Definition

When writing a news piece, content creators often search for related content in order to enrich or offer background to the news pieces they want to develop, so that the readers get a more detailed and informed experience, enhancing the quality of the news pieces. An example of related content could be past events of certain topics or news that are similar to the ones presented in the main news article.

News articles are released every day on the internet, through news channels, newspapers and other types of news aggregators. Despite the fact that these news sources provide content creators with a substantial collection of news which benefits the search of related news content, challenges arise regarding the exploration and navigation through these

[1] How Many Stories do Newspapers Publish Per Day? https://www.theatlantic.com/technology/archive/2016/05/how-many-stories-do-newspapers-publish-per-day/483845/ last access: 2021

large sets of data. The amount of information might be overwhelming and effectively filtering through in order to achieve the desired related content might become a difficult task. The content creator might also be in an exploration mode, learning about a specific topic. For that reason, a search query can be generic at the beginning, but as the user gains insight through the exploration of news collection results, subsequent queries get further refined which could help content creators have a better understanding of what they should search so that they can filter the news to receive the desired results. To achieve this goal, there is a need for tools that allow users to explore these news collections in an effective and turn-based manner.

This thesis proposes an interface to explore collections of news through the use of data visualization techniques. The system was built on top of a machine learning module being developed in the same project in a dissertation by Cláudio Bartolomeu.

1.2 Objective and Proposed Solution

The aim of this dissertation was to design and implement an interactive media-oriented web application interface to help content creators find and then introduce related content in their developing news piece. To achieve this goal, we need to provide the user with tools that allow him/her to efficiently explore large collections of news so that the user can find the desired related content more effectively than current used methods. A data set [71] from the New York Times news is used as an example of data input. The New York Times data set is composed of images that were previously used in a news piece and text. Regarding the data set text, it is composed of fields such as: the news pieces main text, titles, snippets, topics and others.

Regarding the interface, it communicates with an Embeddings System that during an user session has the purpose of providing news piece recommendations based on users' search queries, in an interactive way.

A multimodal system allows for several distinct means for input and output of data, thus providing the user with multiple types of information searches by interacting with the system [24].

In the context of this dissertation, the user is able to query the system using two media types: text and images. User's search queries are multimodal meaning that the user can query the system by utilizing text (e.g., keywords, topics), images (e.g., using a whole image or a region of interest), or both in one query. Additionally, the developing interface provides means of visualization of the user's search results (text, images or hyperlinks) and allow them to explore these results so that they can gain insight about the searches. The system also allows the swift refinement of the user's query. In order to achieve this refinement, the system provides the user with a turn based exploration of the news collection with each turn being a more concrete and precise search query leading to more desired results. A turn is another query that takes into account all the previous queries.

The developed Data Visualization System explores visual graph visualization, image region of interest analysis, time analysis and other types of visual metaphors to refine the query in a turn-based fashion.

The system uses a JavaScript interface, graphics and data visualization tools while connecting to a search server that provides query results.

1.3 Contributions

The contributions related to the development of this thesis are as follows:

- **Implementation of client-servers system** - This client-servers system is capable of receiving data from multiple sources. The servers communicate with an embeddings system that provides recommendations based on users' searches and also processes client results. It is also responsible of caching or transform data if needed.
- **Development of multiple visualizations** - Several visualizations were implemented to support the search and exploration of news collections conducted by the users such as: graph visualizations, time analysis visualizations, embeddings visualizations.
- **Users' interaction with the system** - User tests were carried for the purpose of understanding how users interact with the system and evaluate its usability and effectiveness at exploring and finding the desired news articles.

These contributions helped us guide the developing work.

1.4 Research Questions

The proposed research in this thesis seeks to answer the following research questions:

Main question - How to use different types of visualizations in order to efficiently and effectively explore large collections of multimodal data, more specifically news collections?

Sub Questions -

1. What is the relation between the amount of information shown in the visualizations and the amount of information perceived by the users? Are visualizations effective at showing the intended information or are they visually cluttered?
2. Is the turn based exploration of news collections effective at filtering information and provide users with more concrete and precise search results each turn?
3. Are timeline visualizations suited to learn how a specific topic unfolds throughout the news? Given that there are relations between entities (e.g. people and countries such as Biden and USA)?

The main question is related to the design and interaction of the Data Visualization System to understand what type of combined visualizations are effective at exploring large collections of news. The following sub questions are related with the usability of the system. The first sub question is intended to understand how visually cluttered the different visualizations are, which in part measures the effectiveness of the Data Visualization System.

The second sub question will evaluate how effective turn based exploration is at filtering information, which takes in consideration the intuitiveness of the whole Data Visualization System and the effectiveness of the data selection and query and fine tunning part of the system.

Lastly, the third sub question will evaluate how suited are the Timeline visualizations at learning how a specific topic unfolds throughout the timeline that that specific topic was relevant.

1.5 Document Structure

This document is composed by six chapters.

- **Chapter 1** - It was given context motivation and problem definition, research questions and the proposed solution.
- **Chapter 2** - In this chapter it is made reference to the related work about different types of visualizations, techniques and modes of interaction.
- **Chapter 3** - The fourth chapter specifies the system requirements, the data used and its structure, the development technologies used followed by a mockup of the interface, the system architecture and lastly, the expected user workflow.
- **Chapter 4** - The fourth chapter covers the different implemented features and visualizations as well as their implementations and workflow.
- **Chapter 5** - The fifth chapter concerns the user tests performed as well as an analysis of the results.
- **Chapter 6** - The last chapter summarizes the work done, highlighting the main conclusions of the dissertation and addressing the future work and possible improvements of the system.

2

Related Work

This chapter presents the related work of this thesis. Section 2.1 focuses on systems that explore large document collections. Section 2.2 introduces systems that use graph visualizations with connected data. Section 2.3 explores dimensionality reduction techniques and ways to visualize embeddings. Section 2.4 introduces systems that allows users to interactively browse collections of images and provide means for users to refine their queries. Lastly, section 2.5 presents visualizations where the use of text is used as a core part of the visualizations in use.

2.1 Exploration of Large Document Collections

With the increase in digital information and information gathering, the amount of data available is vast and tends to keep growing. Document collections are part of these data types. Several approaches intending to visualize document collections in order to offer some overview of its content and allow users to perform some type of exploration have been proposed [35, 31]. It is often the case where a user does not have specific query in mind but rather has some general idea of concepts or topics they want to search. For that reason, the first search that a user conducts might start "general" but as a user explores and gains insight and better understanding of the topic in search, the subsequent searches become more specific.

There are several systems that focus on the concept of exploring large collections of documents, being considered next systems analyzed for the objective in study.

2.1.1 Visual Metaphors - Exploration of Documents

Metaphors in its graphical or visual representation are fundamental to information visualization. Information graphics and other interactive information visualization systems often make use of several metaphorical devices. This is done in order to make abstract, complex and voluminous information or transform difficult to understand information in more understandable in graphical terms [65]. For the reasons previously described, several systems that explore some type of visual metaphor will be analysed.

Newdle [78] is an interactive novel visual analytics system written in C++ that tried to tackle the problem of having to browse extremely large volumes of online news by offering users tools that allows them to effectively and efficiently browse news topics, search news of interest, and detect temporal trends. Newdle focuses on exploring large online news collections when the semantics of the individual news articles have already been tagged. Newdle stands for *News Wordles*, a visual metaphor used by this system. *Wordles* are a web-based tool for visualizing text by creating tag-cloud-like displays that give attention to certain aspects of the words such as color, typography, and composition [72]. The authors assume that because there has been extensive research on automatic entity extraction and document summarization, individual news are perfectly described by their tags, also claiming that it's not hard to visualize untagged news by integrating existing entity extraction and document summarization algorithms.

Newdle is implemented in a way that automatically clusters and creates path analysis on networks composed of articles and tags. In order to visually represent the semantic information of clusters of articles, it uses *wordles* as well as line graphs on top of the *wordles* to give the user an idea of temporal trends of these clusters by revealing the daily number of articles on a certain topic. It also allows the users to visually perform an in-depth analysis on large collections of news based on: topics, a cluster of articles related in content, the system allows users to browse semantic information and temporal trends; Tags, the system provides the users with the possibility of searching articles and topics by tags, searching by tags related to another tag of interest and even comparing two tags regarding the articles and topics both have. Figure 2.1 is an example of an article search based by the tag "Haiti" [78].

Lastly, the system allows users to perform article analysis by searching articles of interest by topics and tags, while also allowing access to related articles of the focused article or articles that share tags with the article at focus.

There are other interesting efforts with the goal of exploring large document collections in a visual fashion. ThemeRiver [28] was one of the first efforts made towards this end. ThemeRiver is a prototype that utilizes a river metaphor for the purpose of visualizing changes in thematic documents temporally arranged. It offers an interactive visualization of thematic changes of large amounts of news documents over time. The "stream" of the river is composed of several themes or topics differently colored, this "stream" increases or decreases in height depending on the strength of the topics that compose the stream as the time line flows horizontally from left to right. Other river visual metaphors were employed in other systems as it happens with NewsLab [25], TextFlow [13] and OpinionFlow [75]. NewsLab for example, allows the user to conduct analysis regarding the variation of themes through time.

2D maps have also been used to spatially represent on maps the contents of document collection. In these types of map visualizations, a spatial metaphor is generally used

where not related documents are placed further away from each other in the map and related documents are clustered together and placed near each other Mark Hall et al [4]. described a novel approach for the purpose of exploring large document collections through a map-based visualization, using hierarchically structured semantic concepts connected to the documents in order to design a visualization of the semantic space similar to a Google Map. Authors argue that they were able to create a map where the higher levels of spatial abstraction have semantic meaning. This work was subjected to user experience evaluation to receive feedback on using the map in an "undirected "exploring" context, to asses intuitiveness, utility, and usability of the map" and the results were very favorable. A figure of the map visualization is presented below (Figure 2.2).

On the left of the figure is presented the initial overview of the map in form of islands where related concepts are clustered together. On the right, a zoomed-in view of individual documents and its clusters.

Celeste Lyn Paul [59] at al. also implemented a map based visualization with the intention of interactively visualize and explore very large unstructured text collections. The system was designed to help users manage information overload by offering progressive visualization such as, clusters of terms, snippets of documents, and others, adjusting the form of the data based on the user's needs while tailoring the data model and subsequent visual to users interests by interpreting their interactions. The exploration of these documents is supported by interacting with the visualization by manipulating the terms in the visualization using semantic interactions. The authors argue that the target users of this system often know little about the documents structure or content and that by helping

Figure 2.1: Newdle: example of an article search based by the tag "Haiti" [78]

the user manage information overload by providing semantic interactions combined with other user interactions (e.g.,zoom, pan) will help the exploration of large collections in a more intuitive and manageable way.

2.1.2 Time-Based Exploration of News

Time-based exploration techniques place a relevant focus on documents date and its distribution throughout time. Next will be presented systems that put a relevant weight in the distribution of news throughout time.

Miloš Krstajic et al. [45] introduced a relevance-based technique to visualize streaming news permitting to see the evolution of news in real time, as the tecnhique "permits to show the stream of news as they enter into the system". The proposed technique is applied to the visualization of news stream using an aggregation system of news that retrieves 80000 reports per day by monitoring over 4000 sites. Figure 2.3 shows the News stream monitoring visualization. The dates at the top show the beginning and ending of the monitoring period and each category and its name is shown on the right. Each news item is represented as a square block of pixels and is located according to its publication date on the x-axis. The color is mapped to the tonality value of the news article, with green meaning high positive tonality, red representing negative tonality and gray represent neutral news, meaning that the news article gives a "positive", "negative" or neutral feedback to the topic in question in the news article.

One advantage of this approach is that it is possible to spot sudden bursts of news articles at a certain point in time, which might mean that a "relevant" event might have occurred, although this is not necessarily true in all cases. It also provides an overview of the news topics throughout time and the "sentiment" of the news sources regarding a news article of a certain topic by its tonality, giving the user a better understanding of the incoming news. Regarding the interaction with the system, it is possible to inspect single data objects which reads the url of the data item and leads to the full text of the article.

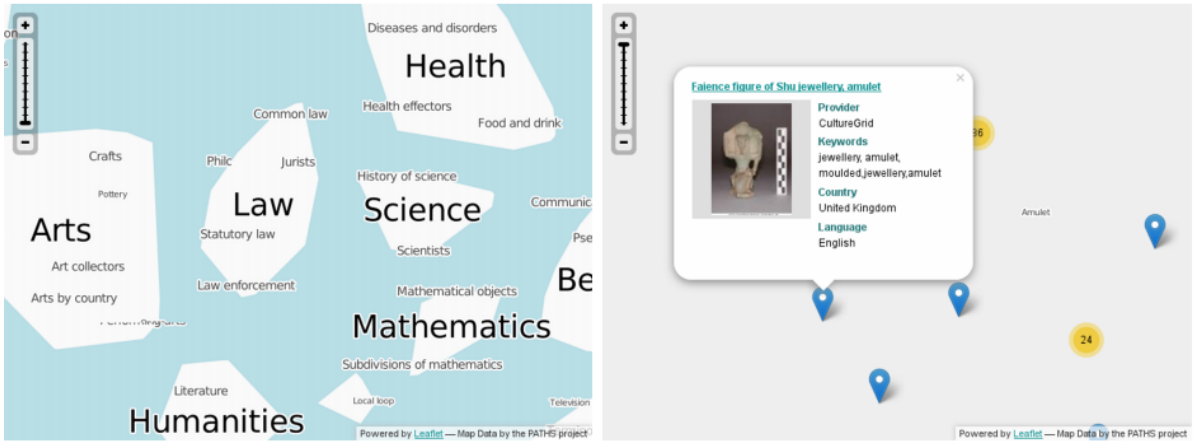


Figure 2.2: image of the map visualization [4]

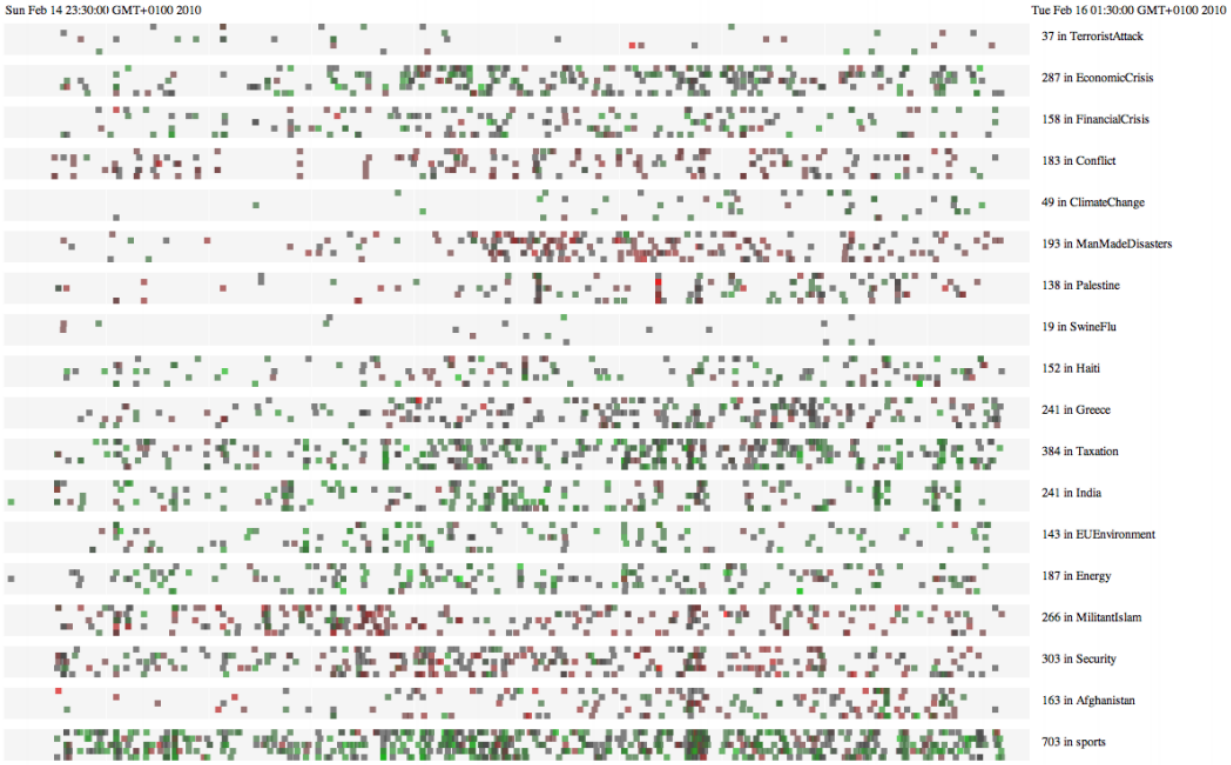


Figure 2.3: News stream monitoring [45]

Tiago Devezas et al. [18] employed three visualization tools for the purpose of exploring a dataset composed of one million news articles from blogs and organizations. The authors use the Media Viz Plataform [19] which is a platform that assists users gaining insight from large collections of news through interactive visualizations. The platform is composed of two main components. A back-end application with the goal of fetching and storing articles while also providing access to the data through an API. Secondly, a client side application written in JavaScript that retrieves the stored data provided by the API and also allows its exploration based on interactive visualization tools. The architecture of the system is similar to the one proposed in this dissertation. The client side application will provide the visualizations and a back-end server will provide the client side with the data needed for the visualizations. The technology stack of the visualizations is composed of two libraries D3.js [10] and C3.js [1]. C3.js abstracts some of the code of D3.js, making it less verbose and easier to use.

Regarding the visualizations provided by the system, users can insert multiple search terms as keywords in order to have a notion of how many articles with those keywords were published in a day, during a selected time period. If a user clicks on one of the data points in the view, it displays a list of all related articles that includes the content of the news, such as a summary, publication date, title or the source's name. Figure 2.4 shows a visualization of the percentage of articles published during September 1st 2015

[1] https://github.com/c3js/c3

and September 30th 2015, the terms were chosen due to being relevant global events at the time. As it can be seen, the amount of news surrounding a keyword "peaks" when a certain event happens related to the keyword.

Lastly, in 2019, Alípio Mário Jorge at al [58]. introduced an interactive system to automatically generate temporal narratives based on news collections. "Conta-me Histórias" is an online tool that generates interactive temporal summarization based on search queries [2]. It used a Keyphrase Extraction algorithm [12] to among others things, select relevant headlines over long periods of times. This framework identifies relevant headlines and important periods over any dataset of news with timestamped data.

Figure 2.5 shows an image with a time interval selection for the query "Brexit", which takes in consideration the last 10 years of news in the Portuguese Archive. The red lines represent interval boundaries. In this case, it identified five important time intervals. The blue are and dots highlights the number of news aggregated by date.

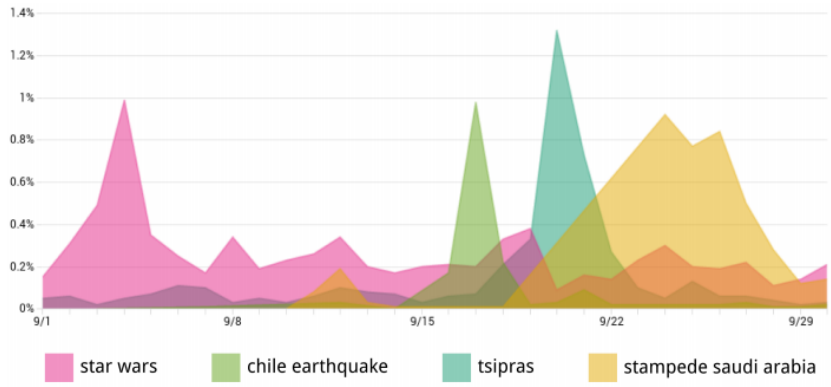


Figure 2.4: Daily percentage of articles published by all news sources containing the given terms in the interface [18]

[2]Information taken from: https://www.dcc.fc.up.pt/site/uploads/files/ECIR-poster.pdf last access: 2021

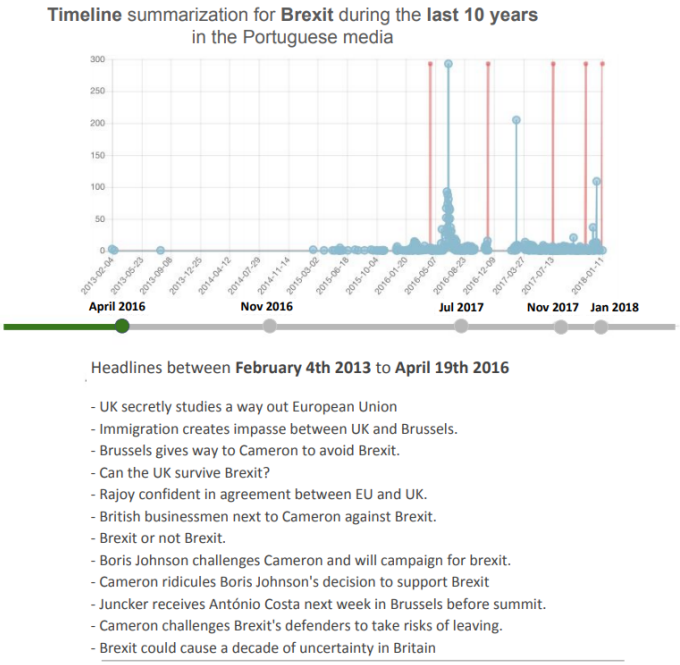


Figure 2.5: Result sample of an interactive timeline [1]

2.1.3 Exploring Large Document Collections

Finding the best way to explore large collections of documents is always a challenge as the amount of information can be overwhelming. Systems that try to tackle this problem often resort to some sort of visual metaphors which aids users exploring these large collections while also providing tools that allows them to analyze collections in a grand scheme so they can gain some insight, and with that, subsequently use that insight to further down users searches and eventually be satisfied with the provided results. For example, Figure 2.2 uses a map metaphor that starts as an overview of the concepts of the collection and to explore those concepts by zooming in on the view revealing individual documents and its clusters. Newdle [78] offers tools to browse topics, search news of interest and detect temporal trends with the use of *wordles* and line graphs while also allowing to perform an in-depth analysis based on topics, related content and others. Those two systems have similar goals, although the metaphors used are completely different.

The systems of Miloš Krstajic et al. [45] Tiago Devezas et al. [18] and Alípio Mário Jorge et al [58]. place a relevant weight in the temporal trends of news. Systems where a user is able to select keywords and choose a time period to visualize the distribution of the news related to those keywords throughout time is extremely important for the purposed work, as a user is able to filter information to a specific time line or gain insight of the different events surrounding a keyword. For example, if throughout the span of five

[1] Figure source: https://www.dcc.fc.up.pt/site/uploads/files/ECIR-poster.pdf last access: 2021

years a specific keyword, "peaks" three times at different time periods in visualizations similar to the Figure 2.4 and Figure 2.5, it is likely that certain global events might have happened. These events might be correlated. For example, in the case of a criminal investigation it is possible for an unresolved case to have been unfolded further at each peak, and for that reason, those three peaks are all correlated, giving the user background about that specific criminal investigation while also creating a timeline of the events. The other case is uncorrelated events. If the word at search is "earthquake", it is likely that those three peaks represent different earthquake events, and are therefore uncorrelated. In both cases the visualization can be beneficial to content creators and provide them with related content for their developing news articles.

2.1.4 Interactive Visualizations of Hypertext Document Structures

Simon Lehmann et al. [46] presents a way to find relevant information in large collections of cross-referenced documents. Highly cross-referenced articles form a directed graph as it happens with Wikipedia. The authors claim that when a user conducts a search query about a certain subject he might not have specific knowledge about that specific subject, and for that reason the user might obtain the desired query result on the first search query, authors also claim that when users search for a subject, they tend to follow references in order to get additional information. With highly cross-referenced document collections, each document gives the user more opportunities to navigate and the structure of the previously visited documents becomes harder to understand.

In order to counter this problem Wivi was created. Wivi allows users to navigate and interactively explore Wikipedia by providing the user with visualizations of the structure of visited articles as well as highlighting relevant topics. Wivi combines these visualizations with a view of the current article results, thus giving the user a chance to explore potencial paths which may lead them to read relevant information with the currently focused.

Wivi was implemented with the purpose of taking advantage of the direct graph created by the articles and the links between them. An example of a the user interface of Wivi is presented in Figure 2.6. While the right side displays the visualization of the article graph the left side displays the article text. The nodes inside the graph represent articles that a user has already seen while the related articles to the already seen are laid out on three rings around the visited articles. An article is assigned to a ring based on a function that determines the relevance of the article, the closer it is the more relevant the article.

Some nodes are highlighted in green to represent the connection between the currently loaded article and its corresponding connected edges (representing the relevant articles of the currently loaded article). Additionally, in the article text, when a user hover over a link, the corresponding nodes and its edges of the related articles become highlighted in blue.

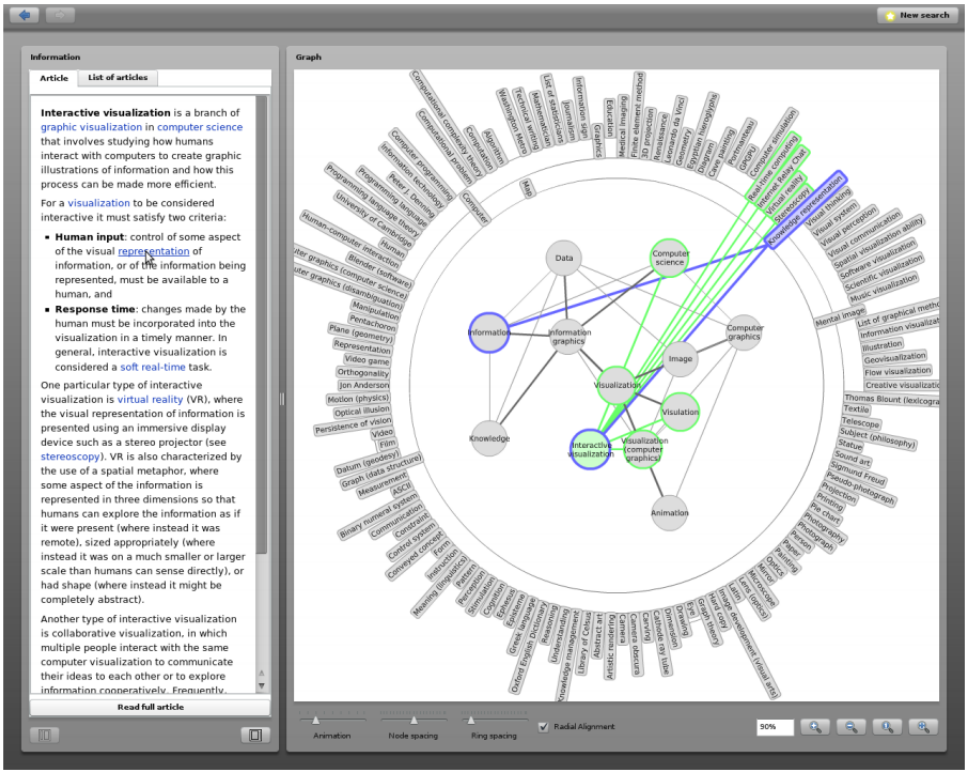


Figure 2.6: User interface of the Wivi system [46]

The visualization and user interface part of the Wivi system was implemented using the Adobe Flex Framework [3] while also using the prefuse flare visualization library [4] Adobe Flash [5] was used for the interactive part of Wivi.

Lastly, the Wivi system was subjected to evaluation of participants, and it was concluded that the approach was generally positively accepted and viably perceived.

2.2 Graph Based Visualizations

The goal of this section is to present different ways of interacting and using graph-based visualizations.

Graph based visualizations are represented by nodes and edges. Often a node is represented by an entity and the edges represent some sort of connections between nodes. Moreover, graph based visualizations are often used for the intent of representing hierarchical structures, as hierarchical techniques have been extensively used in large graphs visualizations, where a graph is continuously being decomposed into smaller sub-graphs forming a hierarchy [8]. However, graphs can also be used with other intents, such as navigating and exploring large repositories as attempted by Christian Hirsch et al. [32] which

[3] http://www.adobe.com/de/products/flex/. last access: 2021
[4] http://flare.prefuse.org/. last access: 2021
[5] http://www.adobe.com/de/products/ flashplayer/. last access: 2021

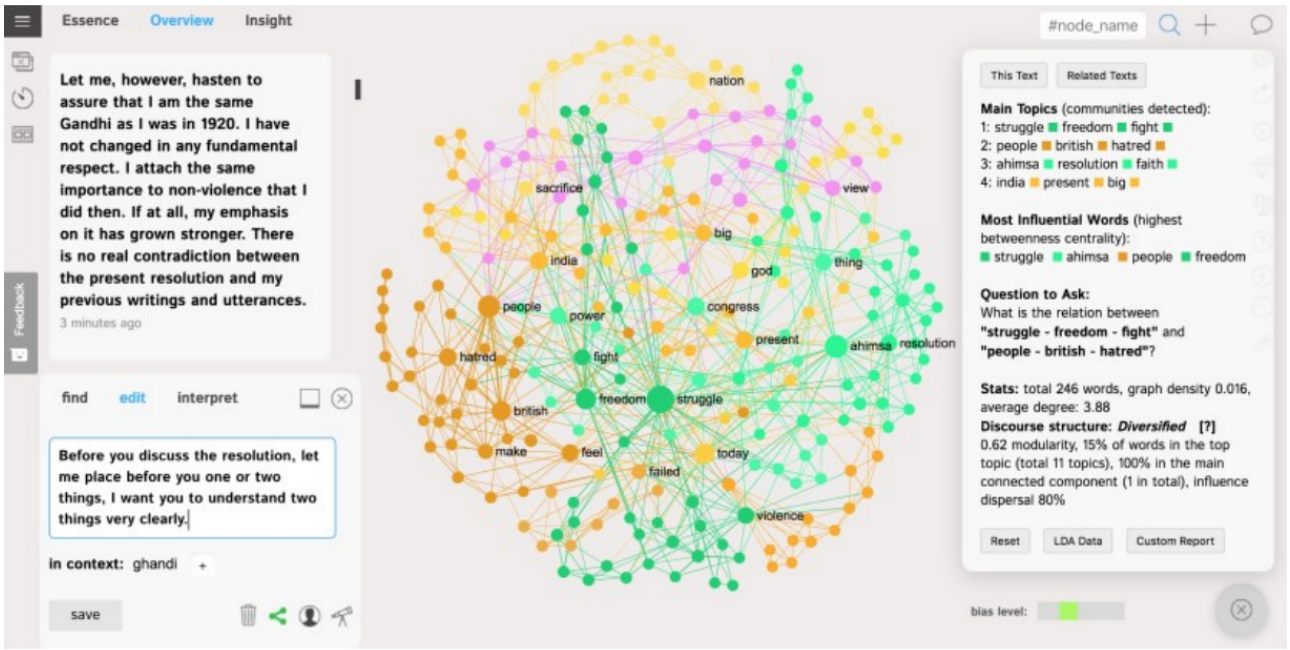


Figure 2.7: Example of the InfraNodus interface [57]

extracted semantically enriched contents from large knowledge spaces (e.g.,wikipedia) to create an interactive graph-based representation out of it.

InfraNodus, developed by Dmitry Paranyushkin [57]. is an open source web-based tool that uses interactive graph based visualizations with the purpose of generating insight from text or discourse by applying text work analysis. InfraNodus was written in JavaScript, using Sigma.Js [6], Cytoscape [7] and other graphology libraries for its front-end implementation. The InfraNodus interface is composed of different elements as seen in Figure 2.7. In the center is displayed a visual network representation of text, the size of the nodes are proportional to the number of connected edges. The color of the nodes is based on the community that it belongs, communities are a "group of nodes that are more densely together than with the rest of the network". In the top left side of the figure lays all the documents that are stored on the system. When a user clicks on a node that represents a word, the system filters the documents to only show the ones with the word presented in it while at the same time only highlighting the nodes that are connected to the selected node, this process can be repeated until no nodes are left or a user is satisfied with the words selected. Additionally, a user can also deselect nodes.

InfraNodus represents a system that combines color, size (node) and graphs gathered with an interactive interface providing users with an understandable way of filtering document collections based on text.

Jennifer J. Pokorny et al. [61] presented a novel manner of visualizing the coding of qualitative data that enables the analysis of the connections between codes using graph

[6] JS library - http://sigmajs.org/ last access: 2021
[7] JS library - https://js.cytoscape.org/ last access: 2021

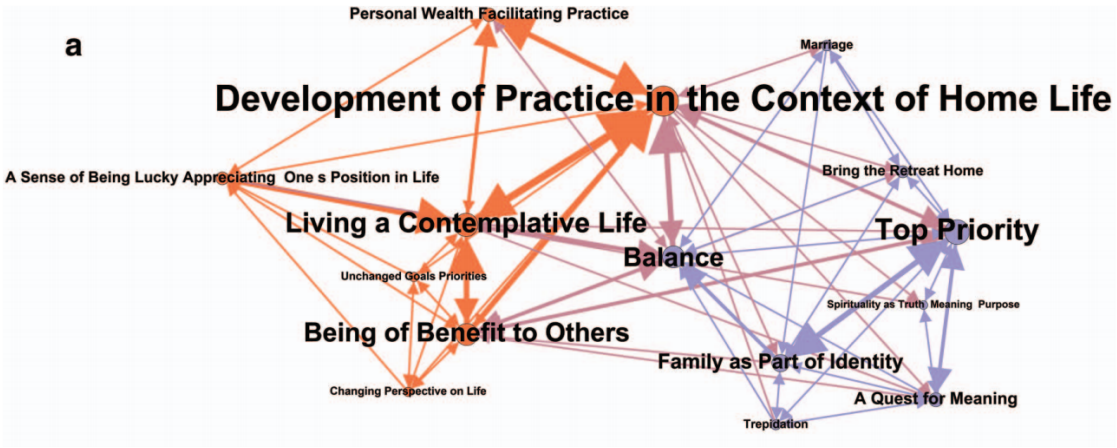


Figure 2.8: Example of a network of the system [61]

theory and network analysis. Using a set of interview transcripts that had been previously coded, network graphs were created from codes that were then applied to a audio file or transcript by using code names and their chronological location, which results in a network that represents the coding data that characterizes the interrelations of codes. Codes are a form of indexing for texts. A code takes the form of a label and is attached to a section of text. This way, it is possible to indicate what information can be found in the section of coded text. Authors argue that this approach will facilitate the examination of associations of network indices with other quantitative variables.

Figure 2.8 shows an example of a created network. The size of the nodes corresponds to its weighted degree, while the color of the node corresponds to assigned community, with nodes of the same color belonging to the same community. The layout of the networks was determined with the use of an algorithm that might run continuously and operates such that nodes repel one another while edges pull their nodes towards one another. Moreover, the order of the codes applied in the text, was taken in consideration and represented as directed edges in the network.

A different kind of graph visualization that combines images and graph based visualizations with the purpose visualizing online social networks has already been presented in 2005 by Jeffrey Heer and danah boyd with vizster [29]. The goal was to build a visualization system that users of social networking services could use to discover and increase awareness of their online community. With this is mind, the authors wanted to support exploration of said networks while also giving users access to search and group patterns.

The nodes of the visualization represent members while the links represent friendship between members, members are represented using both their name and photograph. Networks are egocentric, meaning that networks consist of an individual and their friends. Although it users can expand other immediate friends by selecting the nodes making them visible as well.

Regarding networking exploration. When the mouse hovers over a node it highlights

that person's friends, and visible friends-of-friends where different colors are used to represent network distance (Figure 2.9). Lastly, Vizster also allows keyword search of the visualized network (Figure 2.10). A search box allows for users to type search queries that filter through members profile information, as shown in the right side of the Figure 2.11. If there is a match between the text and the query, an "aura" around the matching nodes are shown (Figure 2.10).

2.3 Embedding-Based Visualizations

"Embedding models map high-dimensional discrete objects into lower-dimensional continuous vector spaces such that the vectors of related objects are located close together" [9]. Embeddings have become popular in machine learning applications because their structure effectively captures domain-specific semantics, even though the individual dimensions and structure of the embedding spaces can be hard arduous to interpret [9]. For example, in 2013, a work conducted regarding natural language processing (NLP) mapped words into real-valued vectors so that semantically similar words were put close together in a view [55].

Visualizations can take advantage of these domain-specific semantics that embeddings can offer. For example, if out of 1000 photos which includes boats, dogs and flowers, a system is capable of semantically separate them in clusters and present them in a view composed by all the images where similar images have similar coordinates, it can give the user a relevant way to show or filter information, because similar images will be clustered together. Figure I.2 depicts the example described.

2.3.1 Dimensionality Reduction Techniques

The proposal Data Visualization System has to communicate with an embeddings system. This embeddings system may work with data that has hundreds of dimensions. Since visualizations must have 2D or 3D dimensions it means that the data sent by the embeddings has to be reduced to either 2D or 3D dimensions [41]. Although dimensionality reduction techniques reduces the dimensions of the original data, and with that comes some loss of information, it aims to keep the "relevant" information of the original data as intact as possible [41]. With this, it is possible to preserve the relationship among different points in the original data set while reducing its dimensionality which can then be visualized and used to explore data and its structure, with the use of clustering algorithms, for example.

Several dimensionality reductions techniques and further comparisons will be presented below:

- **Multidimensional Scaling** - Multidimensional Scaling or MDS [70] intends to find the best subspace that maintains the inter-point distances using linear algebra operations. For example, an image has an similarity matrix that corresponds to its

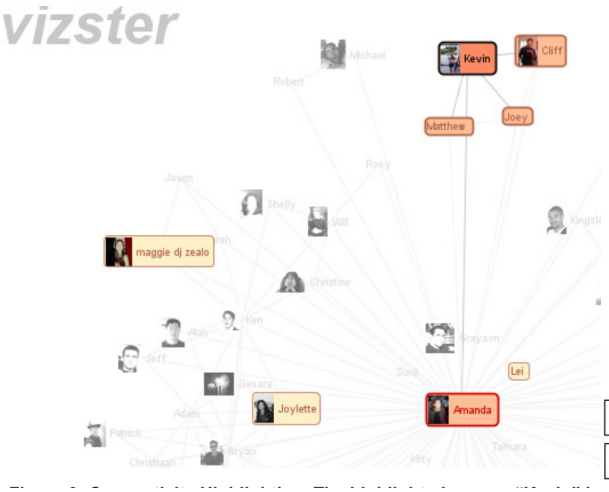


Figure 2.9: Highlighting of a persons friends and friends of friends by mouse hovering over person [29]

Figure 2.10: Searching the network for the word "student". Matches are shown by outlining node borders in red [29]

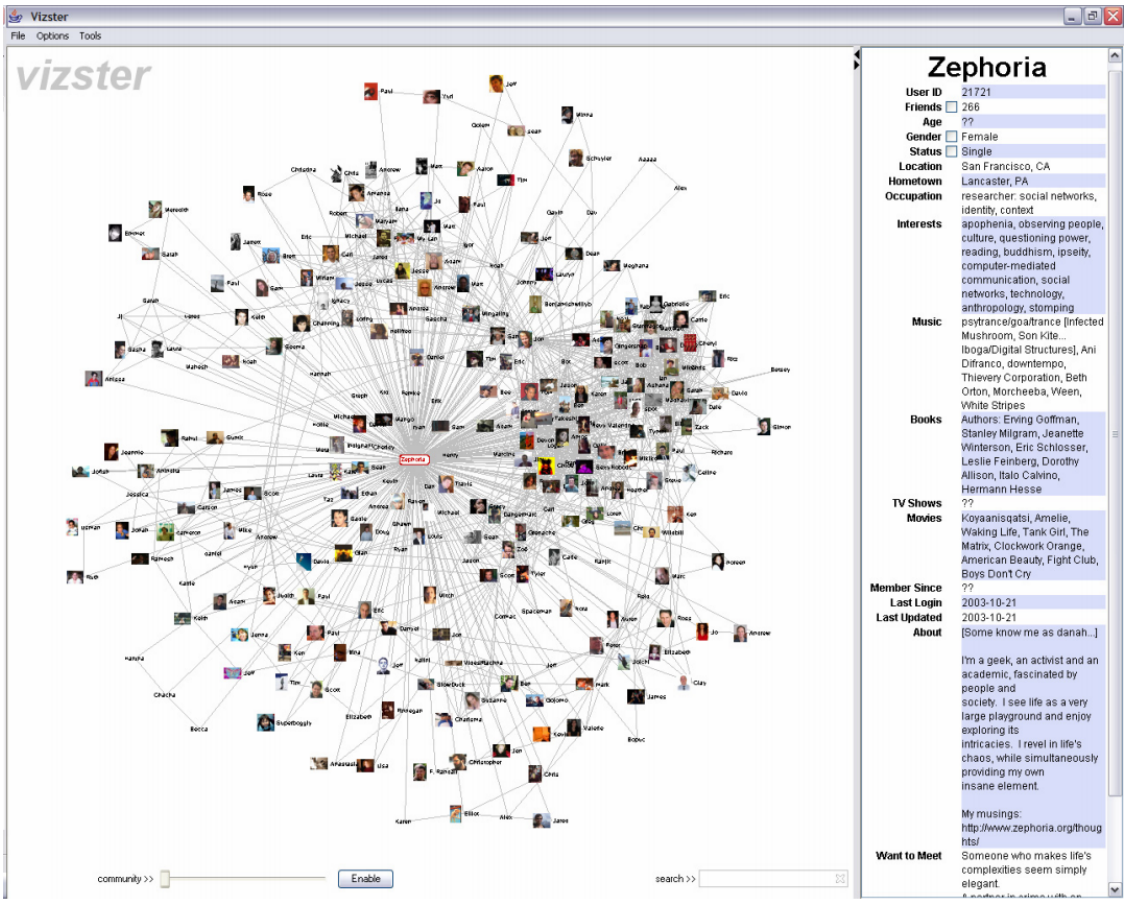


Figure 2.11: An image of the Vizster visualization system. The left side presents a network display with controls for keyword search. The right side presents a panel showing a selected member's profile information. Words in the profile panel that occur in more than one profile will highlight on mouse-over [29]

high-dimensional space. By using the matrix as input, the result is a set of coordinates that represents these images in a low dimensional space [38]. This technique is a linear technique.

- **Principal Component Analysis** - Principal Component analysis or PCA [40] is an Eigenvector intended to model linear variabilities in high-dimensional data by computing the linear projections with the highest variance from the top eigenvectors of the data covariance matrix [38]. This technique is also a linear technique.

- **t-Distributed Stochastic Neighbor Embedding** - t-Distributed Stochastic Neighbor Embedding or t-SNE [53] is a non-linear technique that works in different ways than the ones previously mentioned. Imagining that a data set is composed of points of high dimensions and we want to lower its dimension to a simpler one, while also preserving the structure of the points and the relation between neighbours. To obtain the probability of all points, t-SNE converts the Euclidean distance between data points into probabilities and then uses a method that retains the structure of the data in the low dimensional map. With this technique, it is achievable a considerable reduction of dimensions without losing information [79].

- **Uniform Manifold Approximation and Projection** - More recently, in 2018, a new technique was introduced. Uniform Manifold Approximation and Projection or UMAP [54]. UMAP is a scalable algorithm similar to t-SNE regarding its visualization quality, but arguably preserves more of the global structure with while also having superior run time performance [54]. UMAP tries to preserve local neighbourhoods in a dataset for each data point in the data set. UMAP identifies a number of nearest neighbours and creates a weighted graph where the nearest neighbours are weighted more heavily. The objective is to then find a low-dimensional representation that preserves the neighbourhoods as much as possible [20].

2.3.2 Embedding Projector: Interactive Visualization of Embeddings

In 2016 Daniel Smilkov et al. [68] presented an embedding projector. A tool that allows users to interactively visualize and perform interpretation of embeddings. As explained before, embeddings often have hundreds of dimensions and an essential step is to reduce its dimensionality to two or three dimensions [68]. In the work presented the embeddings are subjected to dimensionality reduction which projects points to a 2D or 3D space, with each result representing a point in the visualization space(Figure 2.12).

The Embedding Projector offers three dimensionality reduction methods, each of these methods can be used to create a two or three dimensional visualizations. Those methods are: Principal Component Analysis (PCA), with the Embedding Projector computing the 10 principal components, t-SNE and custom.

Regarding the interactions with the visualizations, in order to explore a data set, users can zoom, rotate as well as alternate between a 2D and 3D view. The 3D view uses several depth cues, for example, fog to fade out distant points or changing the size of points based on the distance to the camera. It is also possible for users to select only a subset of points or the nearest neighbors of a given point and perform t-SNE or PCA only on those points in a restricted view. Moreover, when there is a data point associated with an image it is possible to simply use the image as seen in Figure 2.14 or use a view of the word embeddings (Figure 2.13).

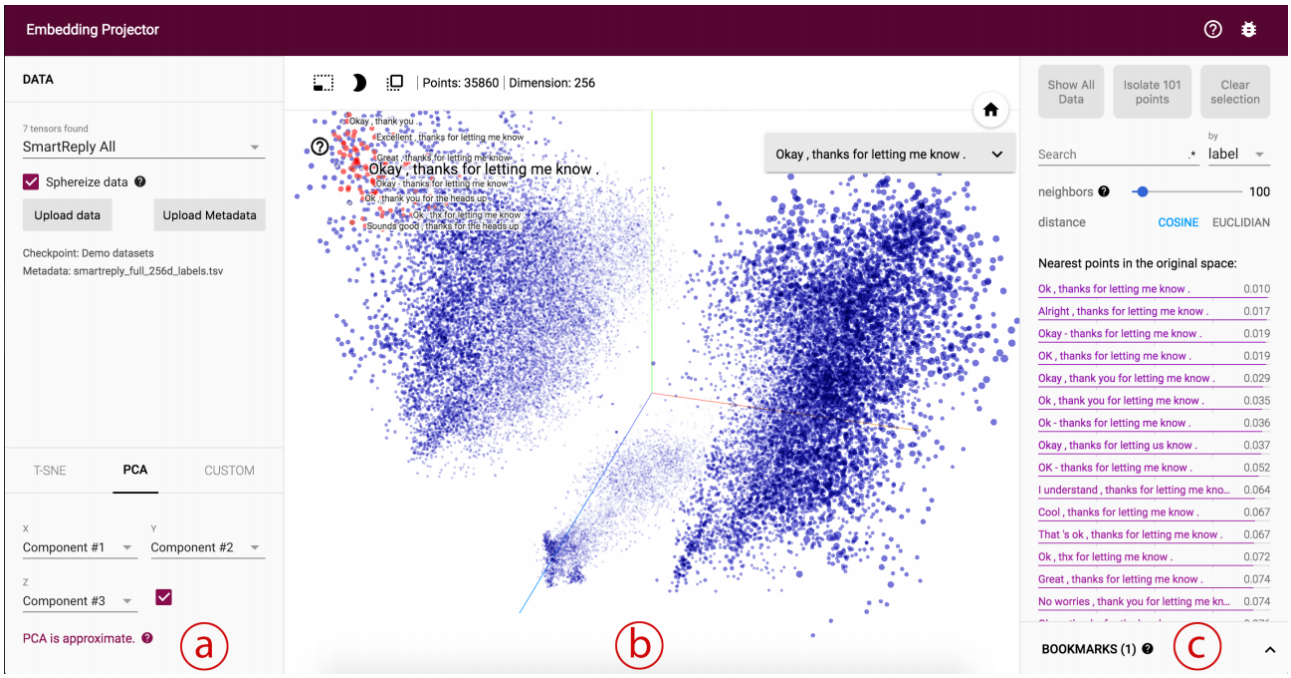


Figure 2.12: A PCA projection of a corpus of 35k frequently used phrases in emails [68]

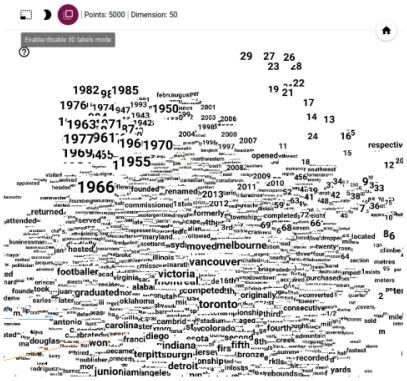


Figure 2.13: 3D labels view of word embeddings [68]

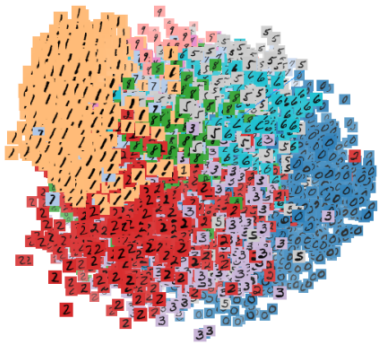


Figure 2.14: View of the word embeddings [68]

2.3.3 Other Embeddings Visualizations

Due to the utility that embeddings can offer regarding semantically structuring large amount of data samples, there has been other works that focus exactly in trying to show visualizations that take advantage of embeddings. For example, Vec2graph [42] is a Python library that focus in visualizing word embeddings as graphs. "Vec" because in distributional word embeddings models, every word is represented with a dense float vector and "graph" because those values are then used to create a graph. Authors affirm that users can easily and effectively visualize semantic relations between words in the form of graphs with any pre-trained word embedding model and the Vec2graph library. Figure 2.15 shows an example of the output for the word "science" and its nearest neighbors in an embedding model. Vertices of a graph correspond to words, while edges indicate semantic similarity between them.

Embeddings can also be used to explore document collections. Cite2vec [6] "allows the user to dynamically explore and browse documents via how other documents use them, information that we capture through citation contexts in a document collection". The user can then dynamically steer documents projections by searching for semantic concepts. Their interface allows users to perform document usage exploration via user-prescribed concepts. Figure 2.16 shows a 2D projections of words, where each word is composed by the phrases "object detection" and "pedestrian". Each document is shown as a disc and is positioned near words whose phrase best describes it usage.

Embeddings can also provide ways to visualize images which then can be used to create visualizations that depend on images, such as spatially-aware representation of semantically similar images. Xintong Han et al. [27] used embeddings to create an automatic spatially-aware fashion concept discovery. Authors then used the t-SNE technique to create a spatially-aware visualization such as the one presented in the Figure I.1 that illustrates the embedding corresponding to the attributes describing colors for tops. Tops with different colors are well separated in the embedding subspace as clusters. These types of clusters can be used as an effective way of filtering as users are spatially aware of the different clusters.

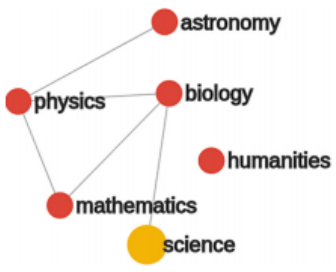


Figure 2.15: Graph resulting from the word science [42]

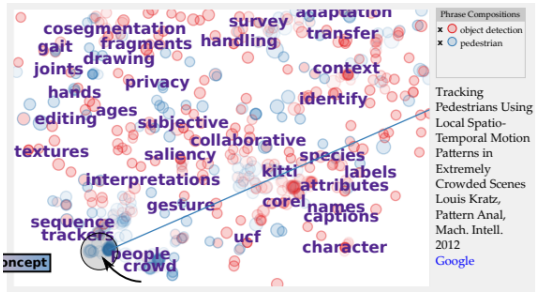


Figure 2.16: 2D projection of the Cite2vec system [6]

2.4 Image Browsing and Query Refinement

One of the goals of this thesis was to create a system that allows users to interactively browse collections of images and provide means for users to refine their queries. News collections have images embedded in the news articles and content creators might be interested in finding news related to some image or specific parts of an image, and thus refine a query in a "visual" fashion, leading a system to only show results based on the content of the image. Additionally, if a system combines visual refinement which takes in consideration the contents of an image with text refinement, with the use keywords, topics or others concepts, it is possible to achieve a more specific query by combining each other strengths, leading to more precise and accurate search results increasing search results satisfaction. There were made several advances towards this end.

In 2006 Jing Yang et al. [77] implemented a SIB (Semantic Image Browser) system that automatically detects semantic image content while also providing several views for image browsing. Using a Multi-Dimensional Scaling (MDS) algorithm it was possible to create a visualization that placed similar miniature images closer together and dissimilar further from each other.

A similar effort was performed with another system in 2005 by M. Dontcheva et al. [22] In this system the authors create a layout that clusters images based on similar metadata and also provides users with mechanisms to interact with the metadata clusters, such as image removal techniques, to facilitate query refinement.

An interesting approach for the purpose of a more accurate query refinement was made by E. Kim et al. [44] The work was conducted for the medical field in order to offer educational or clinical users an effective content-based image retrieval method. The system is able to perform queries based on specific regions of an image, not needing the whole image itself. Furthermore, it is also possible to refine the query using text as the system provides several text cues related to the image such as caption, outcome, title and others. The keywords of this text can be selected leading the system to perform a binary match to other associated image text resulting in the narrowing of search results and a more accurate query. Towards this end, the authors built two main components: a user interface that allows the user to select a specific region of interest in a query image, and a method to decide what regions within their database are relevant and compare the query regions with the database regions. Figure 2.17 shows an image of a manual selection region of interest by clicking points around the figure, the selected part can then be used to refine the query search. Figure 2.18 shows an example of a text query refinement by selecting keywords. Users select keywords based on several text cues given related to the image.

Another prototype with the goal of performing an opportunistic search to explore annotated image collections was made by Paul Janecek and Pearl Pu [37]. The authors argue that it is often the case where a user is not sure of a specific query to make and that the query gets refined as a user gathers information. The prototype allows users to query

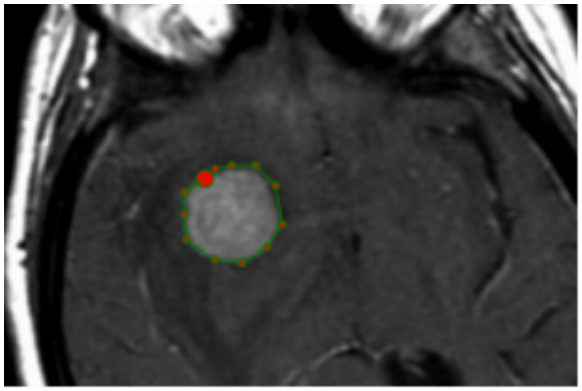


Figure 2.17: Image of a manual selection region of interest by clicking points around the figure [44]

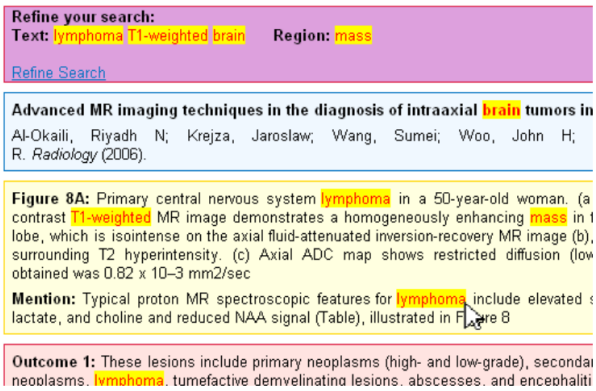


Figure 2.18: An example of a text query refinement by selecting keywords [44]

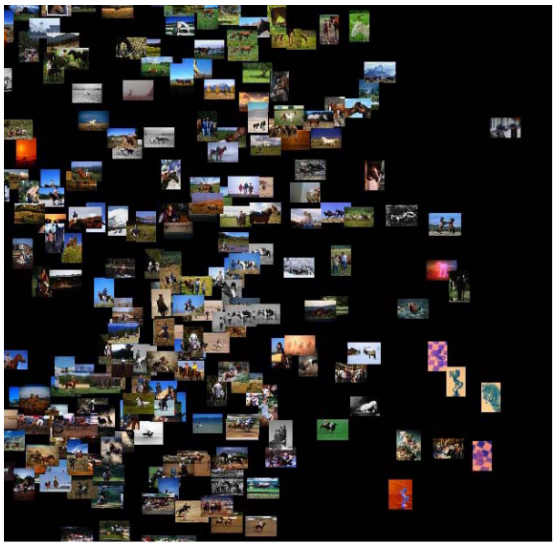


Figure 2.19: Mapping of query results onto a view [37]

Figure 2.20: Highlighted keywords related to the one at focus showing potential directions for future queries [37]

the image collection, mapping the results onto a view, as shown in Figure 2.19. The environment shows a history of previous searches while also allowing users to gather images and keywords. As the user explores the image collection, the view highlights keywords and other images related to the one at focus (fig 2.20), showing potential directions for search that are more specific, general or close to the one at focus.

More recently, in 2019, Xiao Xie et al. [76] developed a system that is capable of interactively visualize large image collections such as an album. The paper proposes a novel visual analytic for the purpose of analyzing images in a semantic manner. The system is composed of two main components: a semantic information extractor based on a convolutional neural network that produces descriptive captions for images, and a visual layout generator that employs a galaxy metaphor turning a projected 2D space to a galaxy

like visualization of images where similar keywords and images are turned into stars and planets. It also allows users to visualize and navigate the provided views in order for users to see a semantic overview of an image collection, and if necessary, perform a more detailed inspection of a certain group of images. The system was implemented with the use of AngularJS for its front-end implementation and a Node.js server for its back-end implementation.

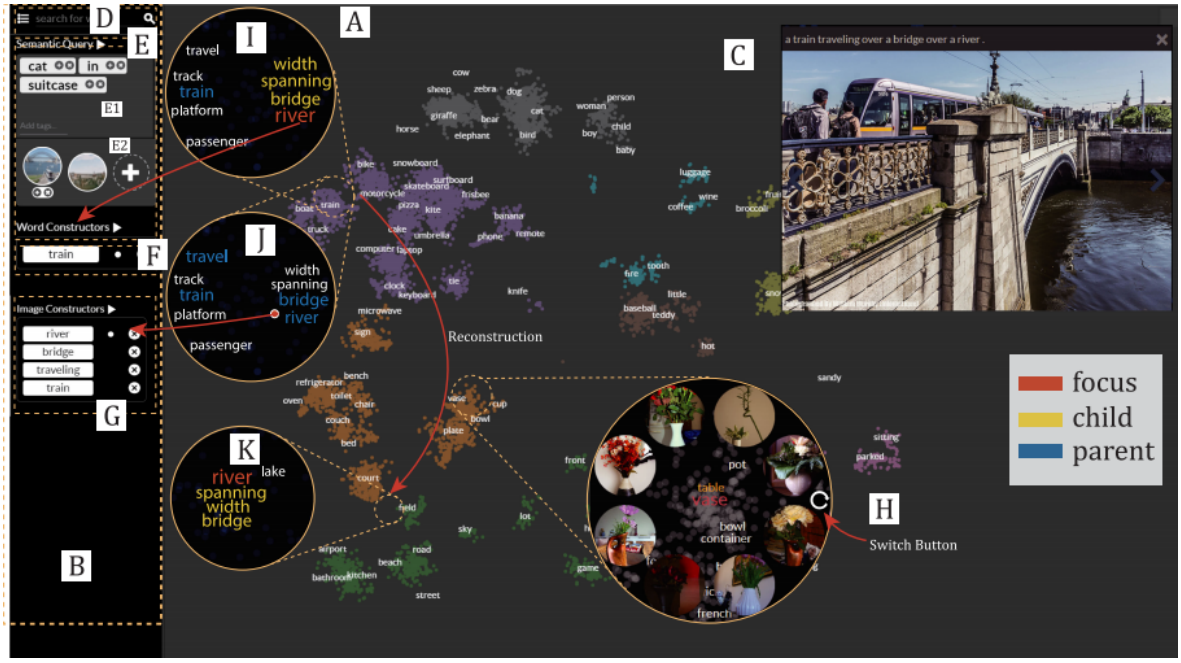


Figure 2.21: Interface of the system [76]

An example of the interface of the system is shown in Figure 2.21 which is composed of several components. The Figure 2.21A represents a galaxy view, this view distributes images according to their similarities and allows users to conduct a multi-scale analysis of images, additionally, a user can zoom into groups of images. The panel presented in Figure 2.21B permits users to perform queries in a flexible manner (Figure 2.213E) based on images (Figure 2.21E2) and tags (Figure 2.21E1) which refines or generalizes query results. It is also possible to search for words presented in the collection (Figure 2.21D), which leads the system to focus and displaying images related to that specific word in a focus-plus-context manner, as seen in Figure 2.21H. Clicking the switch button shows other related images to the ones at focus. When a user performs a query, the resulting images alongside its caption are shown in a browser (Figure 2.21C). The presence of the image caption facilitates the user's work in case the user wants to deviate from the query results or further narrow them down by not using or using the words presented in the caption respectively in the subsequent searches. Lastly, when a user clicks on an image a list of relate words is shown (Figure 2.21G).

The system in question was subjected to user evaluation through questionnaire ratings and user feedback. Overall the user's feedbacks were positive, stating that the system

was easy to learn and the design intuitive and aesthetically pleasing. Although some improvements were proposed regarding the visual design, semantic summarization and semantic layout.

* * *

Early image browsing techniques often took into account their semantic content and placed the images with similar content together and dissimilar further away as it happens with the [77, 37] systems. M. Dontcheva et al. [22] clustered images based on similar metadata, which is a similar technique to the one previously discussed, as both techniques clusters images based on some type of information available about the images.

Regarding query refinement, the most popular strategy of searching and refining that search is, even to this day, through text. Text is an effective method of searching for information, it is the standard way of searching in search engines and many systems that offer some type of searching options. Although text can be an effective type of searching, other strategies have to be taken into account. Other strategies might have some sort of strength that text lacks and vice-versa. If we can combine refining strategies we might combine each others strengths. E. Kim et al. [44] did exactly this, developed a system that was able to perform queries based on images and specific parts of images which returned results (images), based on the content of the searched image and then it was possible to further narrow this search by selecting keywords based on several given text cues related to the image. Xiao Xie et al. [76] implemented a similar system that lets users refine their queries with images and text in the form of tags and words.

In conclusion, although text queries are effective, content based image queries offer the ability to query based on the contents of the image. Sometimes it might be difficult to get the same results with a text description only, as text might not be precise in describing the contents of an image.

2.5 Text Visualization

Several interactive systems have a lot of data in form of text and for that reason, it may seem logical to use it when creating a visualization. But text is a challenging data type to visualize as text is itself a visual encoding and does not vary much [67]. Additionally, J.Bertin [7] affirms that text can vary in color, style, size, and orientation but by doing so it also hampers user readability. There are several approaches made with the intent of using interactive text visualization techniques.

2.5.1 Hyrid Text Visualizations

Hybrid text visualizations combines text and other visualizations as the core of a view.

TIARA [50] (Text Insight via Automated, Responsive Analysis) is an interactive visual text analysis tool with the purpose of aiding users in analyzing large collections of text. TIARA uses a topic modeling engine in order to summarize collections of documents

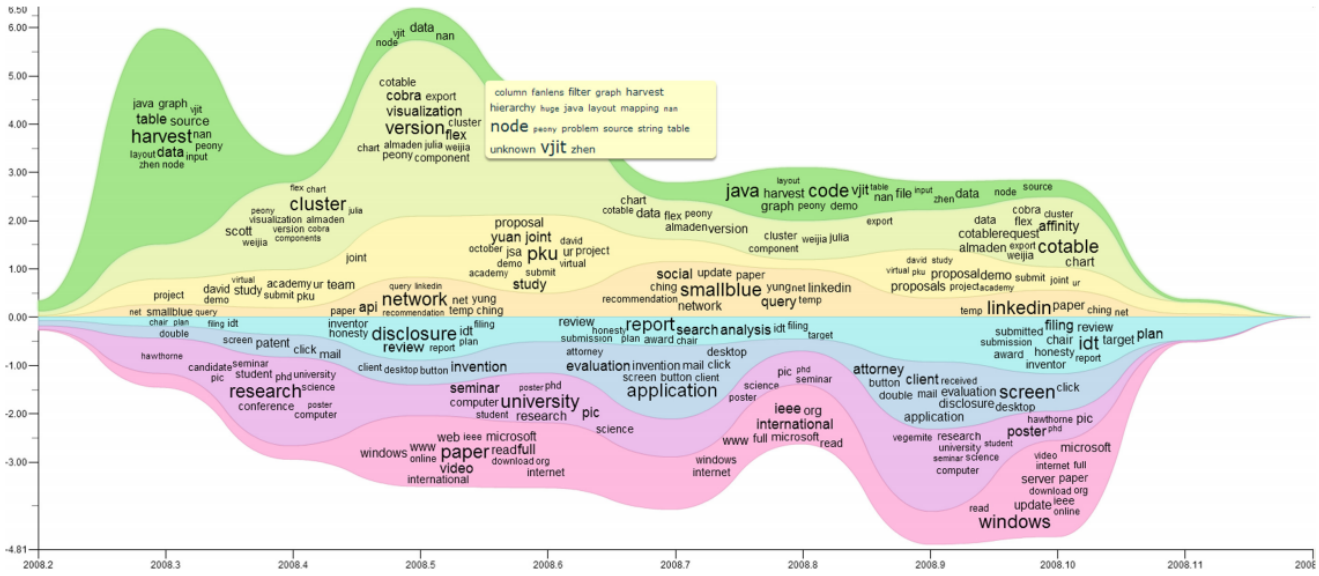


Figure 2.22: A visualization supported by the TIARA system [50]

into a set of topics, each represented by a cloud of keywords, as tag (keywords) clouds can be used to support examination of text collections [73]. Secondly, TIARA uses a time based visualization that shows topic variation over time. Similarly to NewsLab [25] and ThemeRiver [28], presented in section 2.1.1. It also uses a river like metaphor for its visualizations. Figure 2.22 shows a summary of about 10.000 emails. The y axis represents topic strength and the x axis the time. Each colored layer represents a topic, and each layer is composed by a set of keyword clouds that summarizes the topic content over time. Lastly, the width of a layer represents the "strength"of a topic at a specific time. TIARA also allows users to interact with visualizations to further examine summarized text. Preliminary evaluation was conducted demonstrating that TIARA is effective in text analysis tasks. In conclusion, TIARA combines river like metaphors with tag clouds to create a hybrid visualization.

As seen before, text visualizations are often accompanied with other types of "metaphors" either to enrich a visualization or because text is a fundamental part of it. For example, Martin Reckziegel et al. [63] presented a Predominance Tag Map type visualization, with the predominance being in tags, combining maps, colors and text as seen in Figure 2.23. A tag map places tags based on their geographic location [36] and a predominance map "shows which data category is predominant for each geographical entity" [63]. A predominance tag map combines both, meaning that tags are geographically placed based on the predominance for each geographical entity. Figure 2.23 shows an example of this type of map. Each color on the map represents a tree category. Each tag specifies the most predominant category in each zone. Although it is already an understandable visualization with color only, tags could improve the number of distinguishable categories.

Figure 2.23: Image of a predominant tag map [63]

2.5.2 Predominant Text Visualizations

Other types of visualizations use text as solely the core of a visualization. Many times, these type of visualizations intend to focus more on document content for the purpose of obtaining the relations and characteristics of said document. For example, Martin Wattenberg and Fernanda B. Viégas published The Word Tree [74]. Word tree visualizations provide users with the representation of word context and frequency, where the size of the word is usually represented by its frequency in the document. In the cited work the root of the tree is based on a user query, and the branches contain the words or phrases used in the document. Users can also click on a branch expanding additional information. Figure 2.24 shows an example of this type of visualization.

A similar type of visualization that is predominately text based are word clouds (Figure 2.25). Word clouds are typically used as means to summarize text in most systems [30]. World clouds are a cluster of words of different sizes. Generally, the bigger the word, the more often it is mention in a document text, summarizing, this way, the content of a given text or document.

One of the disadvantages of word clouds is that they provide only statistical summary without taking into account linguistic knowledge [30]. There has been made a lot of works that take word clouds into account or a variation of it [39, 51, 15, 14, 17, 56].

Recently, in 2017 Mengdie Hu et al. [33] introduced a novel technique that allows users to visualize content of unstructured social media text. SentenTree, the name of the system, displays frequent sentences patterns using a node-link diagram where nodes are composed of words and the links indicate word co-occurrence in the same sentence, combining text and graph diagrams. SentenTree takes design ideas from word clouds and Word Tree in the sense that the size of the nodes represents word frequency while the spatial arrangement of nodes represents word ordering. Authors argue that this from of visualization may help people gain a fast understanding of key concepts and opinions

of large social media text collection. Although the system was evaluated only by three users, the user feedback was positive.

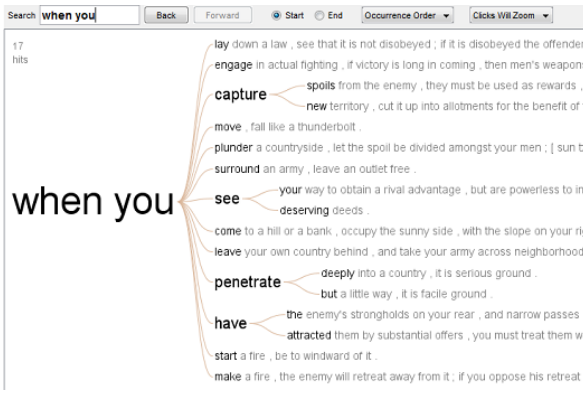


Figure 2.24: Image of a Word Tree[1]

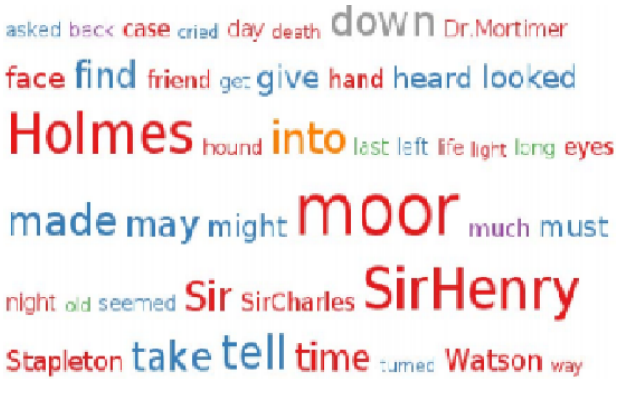


Figure 2.25: Image of a Word Cloud [2]

2.5.3 Critical Summary

Although most data is in the form of text, visualisations that are predominately text based generally lack complexity and are limited to a few. This is not only but also due to the fact that text can only change certain aspects of its structure, such as size, color, font and others. One example of a predominately text based visualization are *wordles*, as previously mentioned. *Wordles* can change in size, color, font, place and others, however, the amount of information that is able to transmit is small and might even be an ineffective visualization as large amounts of text combined with different colors, size, fonts can easily turn out to be an unsatisfactory visualization. For the reasons previously described, generally predominately text based visualizations are best when the goal is to produce an overview of the most relevant topics, keywords or phrases of a collection or similar.

Visualizations that combine some type of metaphor with text, where both the metaphor and the text constitute the core of the visualization, as it happens with the predominant tag map (Figure 2.23) or the TIARA system previously described (Figure 2.22), generally offer a more satisfactory visualization while also granting the opportunity to transmit different types of information, mostly due to the metaphor that accompanies text, as it allows for more abstract and effective ways of showing information. Nevertheless, although visualizations that combine predominately text and other types of metaphors are sometimes effective, they are still not very effective at representing complex information, as text often has a very specific purpose, transmitting a low amount of information which partly leads the visualization to become "hostage" of what the text can represent.

Visualizations with the intent of representing varied and different amounts of information rarely use text as a relevant component and is manly used as form of label, title

[1] Image source: https://sites.google.com/site/textanalysisproject/word-tree last access: 2021

[2] Image source: https://www.computer.org/csdl/pds/api/csdl/proceedings/download-article/12OmNqNG3jl/pdf last access: 2021

or similar, as these visualizations use several components and metaphors combined in order to transmit a large amount of information without becoming too confusing and turning ineffective, as large amounts of text might easily impair some of the metaphors combinations.

3

Media Query and Visualization System

Regarding the third chapter, Section 3.1 introduces the system requirements followed by the used development technologies in Section 3.2. Section 3.3 makes reference to an interface mockup. Section 3.4 presents the system architecture. Lastly, Section 3.6 explains the expected user workflow while interacting with the system.

3.1 System Requirements

The objective of this dissertation was to create a media oriented web application interface to facilitate the introduction of related content in a news piece, so that it is possible for content creators, more specifically content news creators to gather related content for their news pieces. The system uses a news set from the New York Times [71] that contains news dating from 2006 to 2019 as example of data input. This interface also communicates with an Embeddings System that provides recommendations based on an initial input. The interface provides means of visualization of the result (text, images or hyperlinks) and allows the swift refinement of the query. For a user to refine a query it may be necessary for the user to first explore and gain insight about a topic, and as he gains insight, form more specific queries in a turn based fashion leading the user to get more desired results. To achieve the previous described requirements, it is necessary to allow the users to query and explore the data set in a visual manner. This necessity makes the processing and transformation of data essential in order to support and provide the visualisations with the necessary pre-processed data. Due to this, there was a need to implement a JavaScript and Python server that process data to support the visualizations.

In conclusion, the fundamental system requirements are the following:

- **Media Oriented Web Application** - The web application provides means of visualization of the search results, namely news with image and text, and also provides features that allow the filtering and swift refinement of search queries.

- **JavaScript Server** - Several visualizations were implemented to support the search and exploration of news collections. The role of this server is the processing and transformation of data needed to support the different implemented visualizations.

- **Python Server** - Similarly to the JavaScript server, the Python server also processes and transforms data needed to support the different visualizations, specifically the processing of data that the JavaScript server can not do in reasonable time.

3.1.1 Visualization Elements

As previously described, the media oriented web application interface makes use of different types of visualizations in order to allow the user to explore, gain insight and refine his queries to achieve more specific queries in a turn based fashion, leading to more relevant results.

Graph Visualization

The Data Visualization System makes use of visual graph visualizations, as much of the news pieces are correlated between them regarding common addressed topics, concepts, keywords and words. Graph visualizations can be beneficial in representing information that can be linked, such as common topics between different news articles.

Time Analysis Visualization

Time analysis and other types of metaphors to refine the query in a turn-based fashion were also implemented. Time analysis visualizations with the purpose of representing the amount of news about a certain topic throughout time, similar to the system presented in the Figure 2.5 are of extreme utility for the purposed work, as it might be able to create a time-line of events about a certain topic. By having a distributed amount of news about a certain topic throughout time, it might get the user attention about a certain time line as is possible to spot sudden bursts of news articles at a certain point in time, which might mean that a "relevant" event might have occurred.

Embeddings Visualization

The visualization system also makes use of embeddings to offer users a more uncommon but relevant way of visualizing data. The Embeddings visualization translates the content of a news article into a set of values. These values can then be used as coordinates in the visualization, where news articles that are similar in their content are placed closer together. The embeddings were extracted using the news articles dataset.

3.1.2 Interface Components and Features

The interface provides the user with means for the user to visualize the different news articles retrieved by the system. It allows the user to refine his searches based on continuous search queries and other features provided by the interface.

News Feed

In order for the user to be able to gather related content for their news pieces the user has to be able to visualize a vast amount of news pieces in a relatively fast manner and decide if the presented news articles are of his interest to eventually access the full article. It was with this in mind that the interface makes use of a news feed that presents the several news articles summarized in a grid structure, allowing the user to swiftly explore the displayed news articles.

Similar News

As one of the requirements of this thesis was to provide content news creators a way to gather related content for their news pieces, it is important to offer users ways to achieve this objective. One the requirements is to provide users with the ability to return news articles that are similar to a specific news article of the user's interest. However, news articles are comprised of text and its images and the user might be interested in retrieving news that are similar in their content, taking into account both the image and text of a specific news article when searching for similar news. The user might also be simply interested in the visual aspect of a news article and therefore is interested in taking only into account an image of an article in order to return news articles that have similar images to for example, enrich their developing news article with images that are in line with what they are developing. For this reason, one of the requirements was to provide users a way to retrieve similar news to one of the user's interest based on its content or visual aspect.

Also, the ability to query the system based on image region of interest is an interesting feature, this way it is possible to select parts of an image, such as objects presented in an image, or a person and use it to further refine the query and receive results that take into account only the selected image of interest.

Search Component

Since the user will necessarily have to search the system, it is necessary to obtain a component in the interface that allows the user to search according to what he wants to search for, hence the need of this component.

3.2 Development Technologies

This section will discuss the technologies chosen for the development of the system as a whole. Since the system requires a web platform that is responsible for supporting the several visualizations and news presentation modes and since the system also requires the development of processing servers, the technologies are separated into front end and back end.

3.2.1 Front End

One of the objectives of the thesis was the development of a web platform capable of supporting several types of visualizations, news presentation modes, among other features. For this reason, there was the need to create a web platform. With this in mind, the main technologies used in its implementation are described below.

- **JavaScript , HTML5, CSS** - The web application that supports the several visualizations and its interactive elements was created with the use of the JavaScript language, HTML5 and CSS to design the whole interface.

- **React** - The web application was developed using the React library. The library makes use of the JavaScript language and allows the creation of web interfaces. The created interfaces are interconnected through the passage of data and are structured in a hierarchy of components. React also makes use of the Virtual Document Object Mode (V-DOM) which is a concept where the representation of the interface's constituent elements is kept in the memory of the browser that the application uses to execute [1]. With React, it is possible to design simple views for each state in the application, and React will update and render only the right components when the data changes [1]. The fact that React is able to only render certain components when the data changes is immensely advantageous to interactive visualizations that often have many components, as re-rendering all the components when the data changes might not be necessary which results in better use of resources.

  React was used to design and create the user interface due to the fact that it is a popular and mature JavaScript framework with the intent of creating user interfaces, and also due to the previous experience with this framework.

- **React-Bootstrap** - Some of the elements used in the creation of the Web platform interface relied on the React-Bootstrap [1] library. React-Bootstrap replaces the Bootstrap JavaScript (or Bootstrap.js) and it was developed as a re-implementation of the Bootstrap framework without the need of the jQuery library[8]. This library not only provides some CSS styles but also pre-developed ready-to-use components.

[1] https://react-bootstrap.github.io/

- **Prime-React** - Similarly to React-Bootstrap, some components of the react library Prime-React [2] were also used to design and developed the web application. This library offers numerous open source UI components, icons, CSS utilities and also templates.

- **D3.js** - D3.js [3] is a JavaScript library that enables direct manipulation of the *document object model* (DOM) for the purpose of creating various visualizations. With this library, designers selectively bind input data to arbitrary document elements, and then apply transformations to modify or even generate content [10]. This library was designed to display digital data in a dynamic graphical form using HTML, SVG while also allowing the designer great control over the final visual result [80]

- **Three.js** - Three.js is an open-source JavaScript library created in 2010 that makes it possible to create and render 3D scenes in the users browser. Three.js allows users to use their GPU(Graphics Processing Unit) to render the graphics and 3D objects on a canvas in the web browser. Three.js also makes use of WebGL to renders its graphics. However, WebGL is very low-level system that is only capable of drawing points, lines and triangles. Generally, to do a useful WebGL program it requires a good knowledge of it and a lot of code. Three.js offers an easy-to-use API that allows users to create and manipulate 3D objects without the need for in-depth knowledge of WebGL [2].

  This library was used to create a 3D visualization.

3.2.2 Back End

As mentioned earlier in section 3.2, since the system also requires the development of processing servers, there is a need to choose technologies that accelerates their creation and development, namely the framework to develop each of the servers. The goal was to choose frameworks that could accelerate the development and be minimalistic, but at the same time have all the necessary functionalities to not affect the development quality.

- **Express.js** - To develop the JavaScript server the Node.js back end framework Express [4] was used. The reason for this choice lays on the fact that Express follows the REST architectural style and is the most popular JavaScript back end framework [16] with about 56.4 thousands GitHub [5] stars which in turn also means that it has a good community support. Also, with the expanding Node ecosystem libraries such as Express it easy to grow and scale simple servers to functional applications [60]. Lastly, Express is also seen as a minimal and flexible Node.js web application framework that provides many features to facilitate web development [3]. This minimalism and

[2] https://www.primefaces.org/primereact/
[3] https://d3js.org/
[4] https://expressjs.com/
[5] https://github.com/expressjs/express

flexibility provided by Express proved to be quite useful since the main focus of this thesis was not placed on building the different servers and a complex framework would turn out to be time consuming and consequently a draw back.

- **Flask** - The framework Flask [6] was also used to build the Python server. Flask is a back end framework and is a small framework by most standards, small enough to be called a "microframework." Although it is a minimalistic framework it still provides a solid core with the basic services needed to build a server [26]. Lastly, Flask is a very popular web framework with 58.4 thousands GitHub [7] stars as well as a very active community. Flask has a well designed API and is very easy to learn [23]. The choice of this framework was due to the fact that it is an easy-to-learn and very minimalist framework, which, as mentioned before, the main focus of this thesis was not placed on building the different servers and a complex framework would end up being time consuming and, consequently, a hindrance.

3.3 Interface Mockup

Below is presented a prototype of the proposed interface (Figure 3.1). The idea was to separate the query results (left side of the figure) that shows the news, its snippets, titles or similar, and allow the user to see the full article (e.g. See more button) from the results. The right side of the figure is used to show the different visualizations and allow the user to refine the query results by interacting with the different visualizations. It is possible to query the system through the use of text and images or parts of images as shown in the top of the figure.

[6] https://flask.palletsprojects.com/en/2.0.x/
[7] https://github.com/pallets/flask

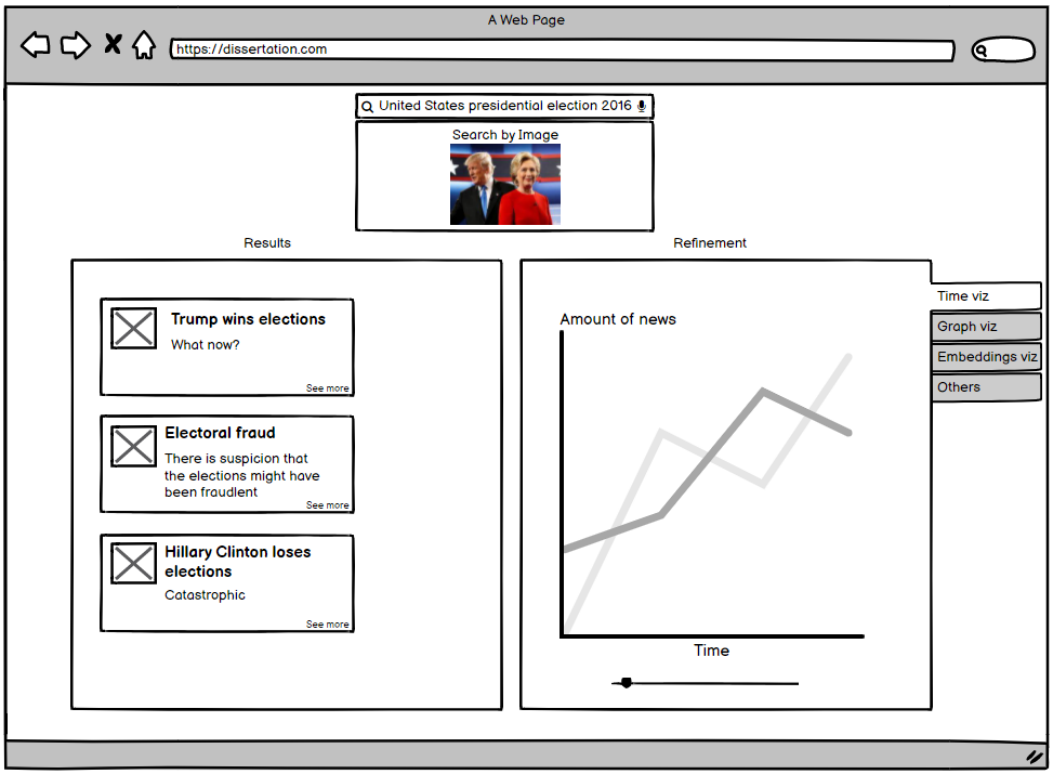


Figure 3.1: Simple sketch of the proposed system prototype

3.4 System Architecture

The architecture of this system is made up of several components that communicate with each other, namely a web platform (front end) that runs the code necessary for whole web page to be up. The user is able to interact with the platform. This platform runs on the user's browser and provides the user with visualizations and tools so that the user can search, explore, filter and refine news articles according to his needs. Figure 3.2 depicts the different components of the system and their interaction with each other.

A JavaScript server serves as the single point of communication for the user, as it receives any request coming from the web platform. This server function is, for example, the transformation and processing of the data required for the visualisations to receive the necessary data in order to be displayed. Most of the processing is done in this server. It also has the role of redirecting certain user requests to other components of the system if a request requires it. The main technology used for the development of this server was Node.js.

A Python server which, similarly to the JavaScript server, has data processing roles, namely the processing of data involving embeddings, algorithms for dimensionality reduction as well as clustering necessary for the 3D visualisation that the interface provides. It also has the role of providing the user with the ability to retrieve similar news articles to a specific one if the user requests it. The Python server makes use of two embeddings

indexes. A visual one and a multi-modal. The visual one is used when it only takes into account the visual content of a news article. The multi-modal one is used when it takes both the visual and textual content of a news article. These indexes group embeddings in a way that embedding similarity searches can be performed.

Initially, the implementation of this server was not foreseen. The creation of this server was necessary because the roles of this server (dimensionality reduction and clustering) were initially implemented by the JavaScript server. However, the processing of these roles implemented in the JavaScript language was tremendously slower. Due to this, and since we realized that the algorithms required to perform the dimensionality reduction and clustering roles were optimized for the Python language, it was decided to create a Python server that implemented these functions. This server communicates solely with the JavaScript server.

The system architecture is also composed by the Elasticsearch component. Elasticsearch [8] is a distributed data search and analysis engine and is responsible for storing all the news articles content and retrieve the actual news articles content if needed. It is possible to query the Elasticsearch with the most complex queries if needed. If the system receives a request that needs to retrieve the actual news articles content, a query is made to the Elasticsearch that responds with the news content. the Elasticsearch can be seen as the database of the system.

The State Aware service is one of the features of the system is the presence of a component composed of an embeddings system that is able to re-rank the news based on previous searches and, this way, provide users with a better ranking of the news based one the user's past searches. The embeddings system uses machine learning techniques in order to provide results the best way possible.

The State Aware service classifies the news through a score between 0 and 1, this score indicates the degree of relevance that a news has in a search, where a high score indicates that the news is very relevant and a low score indicates that the news is not very relevant. This service takes into account previous searches to create a new score. For example, if we consider two cases: in one, the user searches first for "trump"and then for "white house". In another, the user searches only for "white house". Although the search "white house"is present in both cases and the news are the same, the score of each news will be different, this is due to the fact that in the first case the search "white house"takes into account the first search "trump", while in the second case only takes into account the search "white house".

The State Aware service component was implemented in another dissertation parallel to this one and is the only component presented in Figure 3.2 that was not implemented in the course of this thesis.

[8] https://www.elastic.co/pt/what-is/elasticsearch

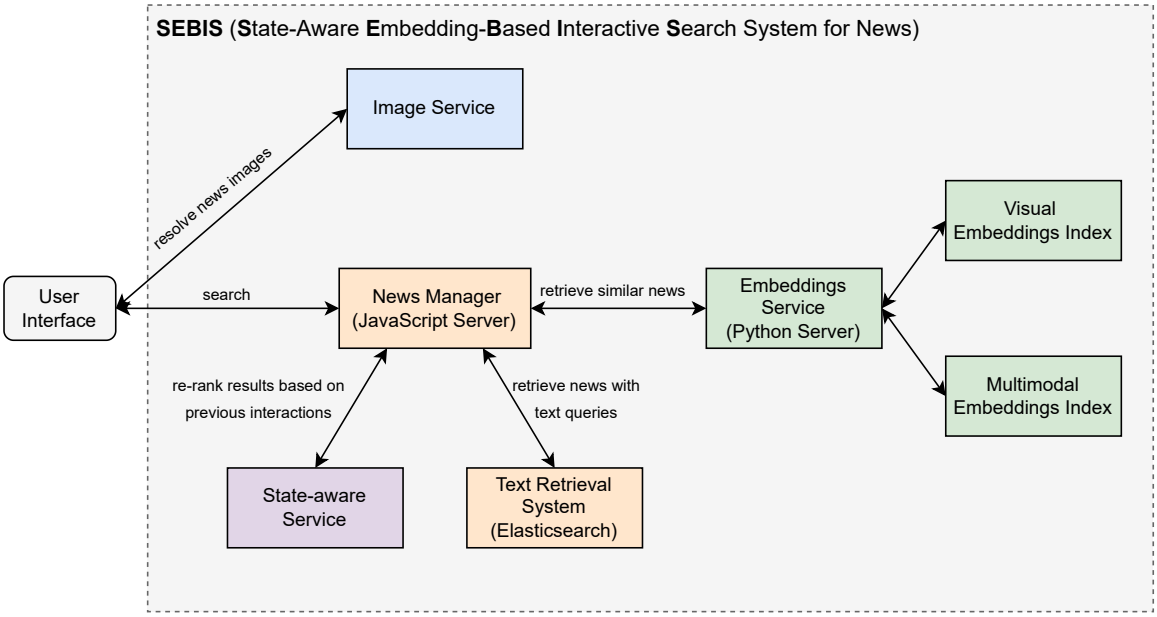


Figure 3.2: System's architecture

3.5 Preliminary Work

In order to initially test some concepts, several small initial prototypes were made to test the concepts. It was implemented a cropping mechanism that grants the user with the ability to make irregular areas in images. This theoretically would be used to allow the user to crop image regions of interest of any image, which could then be used to query a system using the resulting cropped image. Figure 3.3 shows an example of this type of cropping. The original image (top) was cut with the use of the mouse, clicking around the figure to make an irregular figure, resulting in the cropped image at the bottom. The prototype was not further developed past this stage as there were other priorities and thus was not included in the final part of this dissertation's work.

Figure 3.4 shows a 3D representation of embeddings results using dimensionality reduction techniques (t-SNE). This implementation was inspired in the Embedding Projector of section 2.12. This prototype inspired another embeddings visualization which makes use of 3D elements.

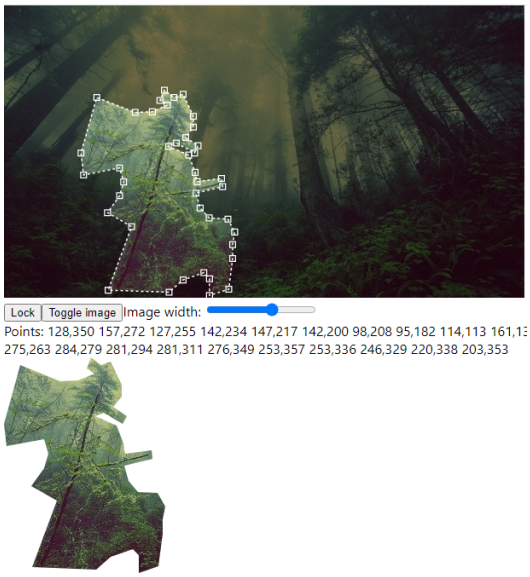


Figure 3.3: Example of the developed cropping mechanism

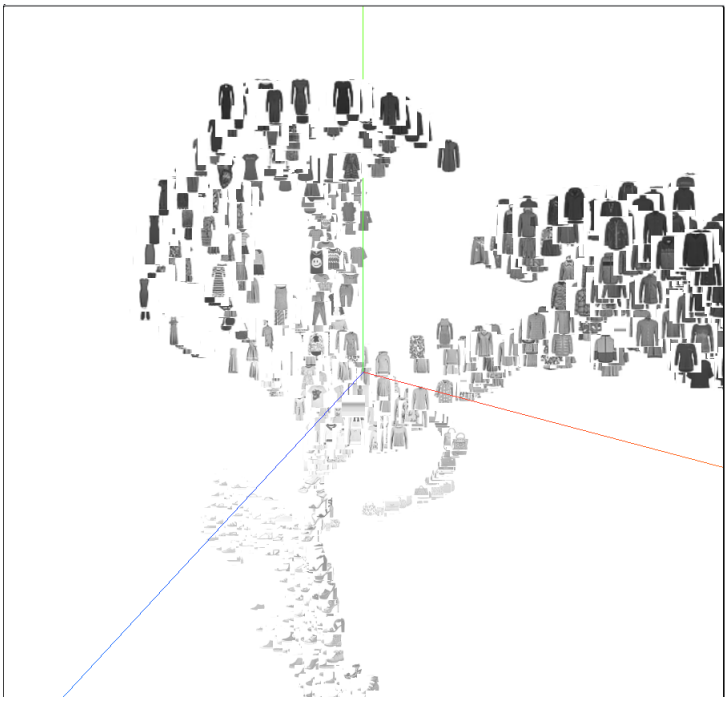


Figure 3.4: Embeddings projector: In this example we can see different types of cloth from dresses to jackets

3.6 Expected User Workflow

The implemented web application was designed with a specific workflow in mind that is expected to be followed by the users. Figure 3.5 depicts this workflow.

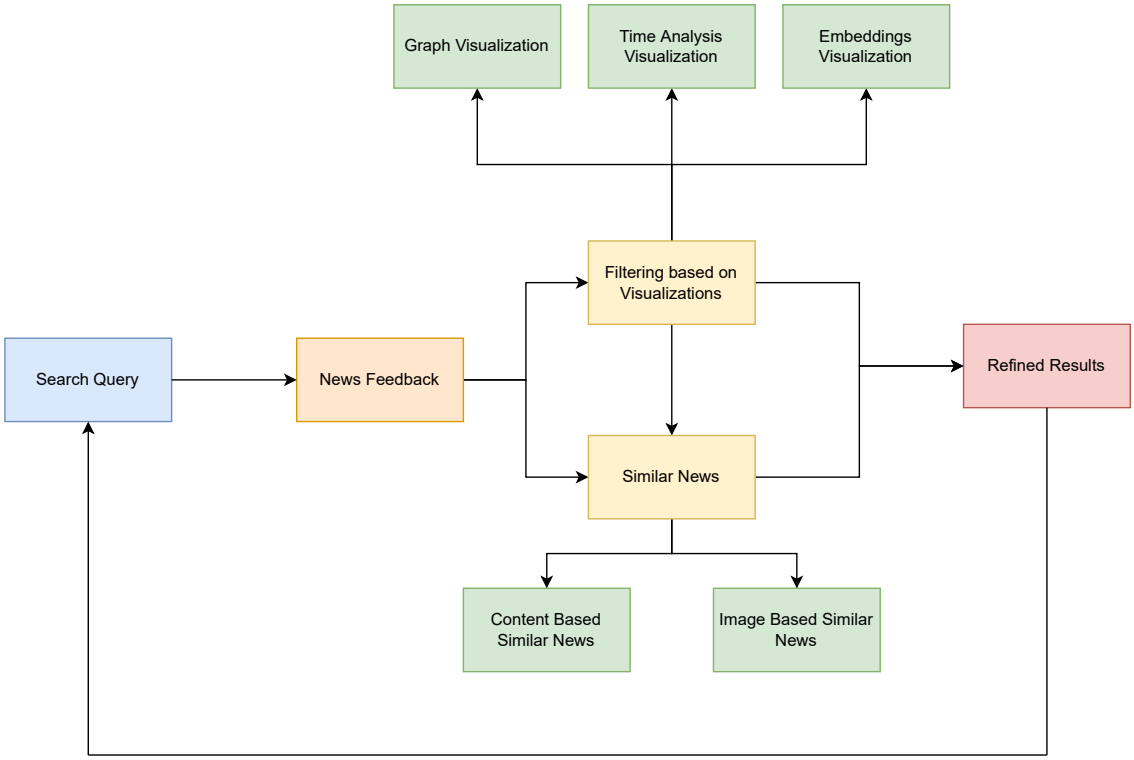


Figure 3.5: Sketch of the expected user workflow

Initially, the user will make a search query based on the his exploration intents. After the system responds with the most relevant news articles based on the user query, the user is expected to explore the news feed and decide if he is pleased with the results. If the user decides that he needs to further explore and filter the amount of news in order to come to the conclusion that he needs to make further queries to get the desired results, he can do so by interacting with the different visualizations, namely: the graph visualization, time analysis visualization and the embeddings visualization. After interacting with the visualizations, if the user decides that he is pleased with the results, the search ends there. If however, he is not completely pleased with the results but found that a news article's content addresses the type of content he is looking for, the user can request the retrieval of similar news articles based on the news article's content or the news article's images. It is important to mention that it is not mandatory to interact with the visualizations before requesting for similar news. If after the completion of these steps the user is still not pleased with the results, he can make additional search queries that take into account all past queries and the whole process is repeated until the user is satisfied with the results.

4

User Interfaces And Implementation

This chapter covers the system overview as well as the implementation and workflow of the different system components, namely the overview and implementation of the different features and views that the system possesses. Section 4.1 gives an overview of the user's interface. Section 4.2 covers the System State and workflow. Section 4.3 covers a feature that allows users to retrieve similar news and their implementation. Sections 4.4, 4.5, 4.6, 4.7, 4.8 address the different visualizations and its implementation, namely: Time Analysis visualization, Embeddings visualization, Similar News visualization, Bar Chart visualization and the Word Graph visualization. Lastly, Section 4.9 covers the storage of the user's session.

4.1 Interface Preview

As previously mentioned, the main goal of this dissertation is to build a web application that allows a user to search and explore news collections through the use of visualizations and features provided by the interface of the web application. Such visualizations consist of graph visualizations, time analysis visualizations, embeddings visualizations and other types of more common visualizations. In this chapter, the different components of the web application and its visualizations will be presented. The type of data that the system consumes and which components of the system the data passes through in order to be transformed or processed will also be described, thus guaranteeing that the data has the correct structure needed for the visualizations to be loaded.

Figure 4.1 shows the interface that the user interacts with separated by rectangles labeled with letters to better explain the interface components. The rectangle in the figure labeled with the letter A represents the interface's search bar which is used by the user to search for a particular topic. One of the most important parts of this interface is the news feed, corresponding to the red rectangle with the letter B in the figure. The news feed is where all the news articles are displayed to the user. The letter C of the figure corresponds to the space where the visualizations are shown and that the user will interact with, making it possible to explore and filter news articles. The space is

displaying the Time Analysis visualization because it is the predefined one. At the top left of the figure (D) three tabs are presented, those tabs are: "Time Analysis", "Similar News", "Embeddings". The name of those tabs corresponds to the function that the visualizations are intended to support. Each tab is composed of one or more visualizations that serve the purpose of helping the user explore topics and/or filter news so that the user can eventually find relevant news articles. The particular details of the visualizations will be explained further in the document.

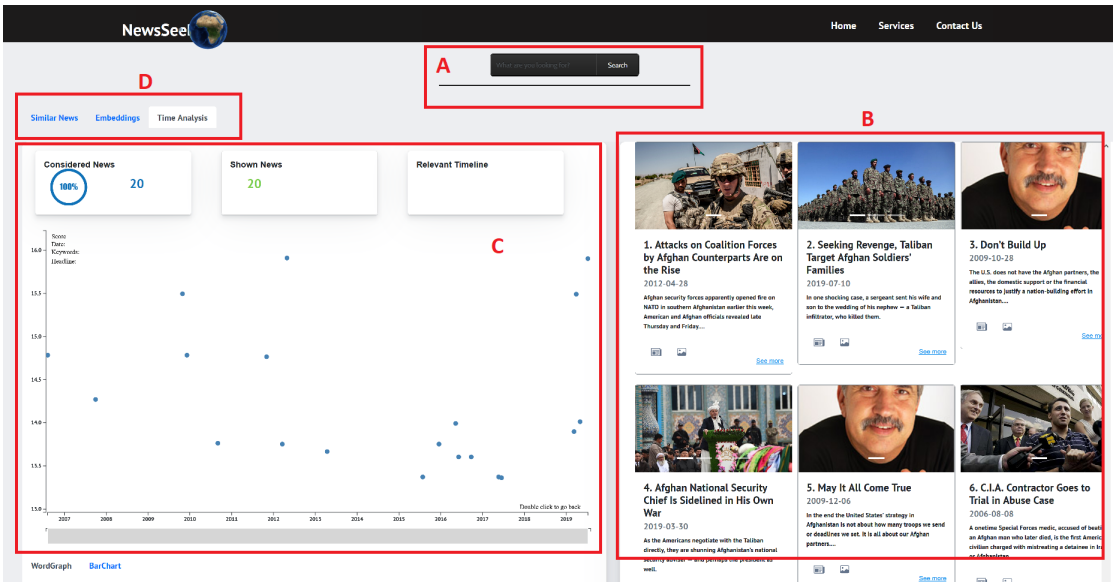


Figure 4.1: Web application interface. The rectangle with the Letter A corresponds to the user's search bar. Letter B corresponds to the news feed. Letter C corresponds to the visualization space. At last, Letter D shows the different visualization tabs.

Figure 4.2 a single news article taken from the news feed. An article is composed of several images, a title, the date it was published and a snippet of the news. If a user decides to access the complete article he can click on the hyperlink "See more", situated at the bottom right of the figure which will then redirect the user to the news article original source. The number of images that a news article possesses is represented by the dashed lines situated at the bottom center of the image of the news article. The specific news article in the figure contains three images, represented by the three dashed lines in the news article image (Figure 4.2). It is possible to scroll trough the images by clicking on each of the extremes of the image, the left extreme to display the previous image and the right extreme to display the next image.

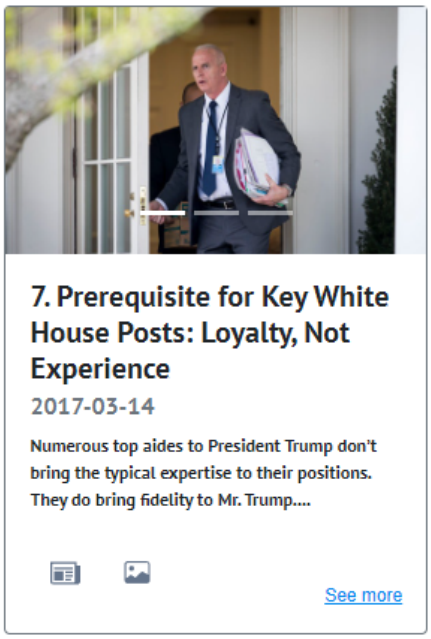


Figure 4.2: News article.

As previously mentioned in section 4.1, the interface also has a search bar at the top which is used by the user to search for a particular topic (Figure 4.3). When a user wants to search for a certain topic he can simply write on the search bar, the system will then return the 200 most relevant news articles of that topic. The relevance of each news is determined by a score obtained by the State Aware System component. The score of each news varies between 0 and 1, the higher the score the more relevant a news article is considered. When the system returns the 200 most relevant news, the news feed is updated showing those same returned news ordered by relevance. Figure 4.4 shows the 6 most relevant news articles returned when searching for the term "Elton John". If the user wishes to explore more news articles he can do so by scrolling through the feed.

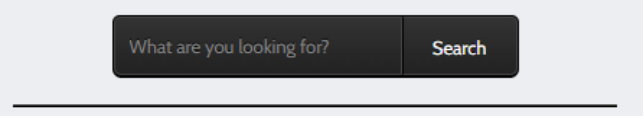


Figure 4.3: Interface search bar

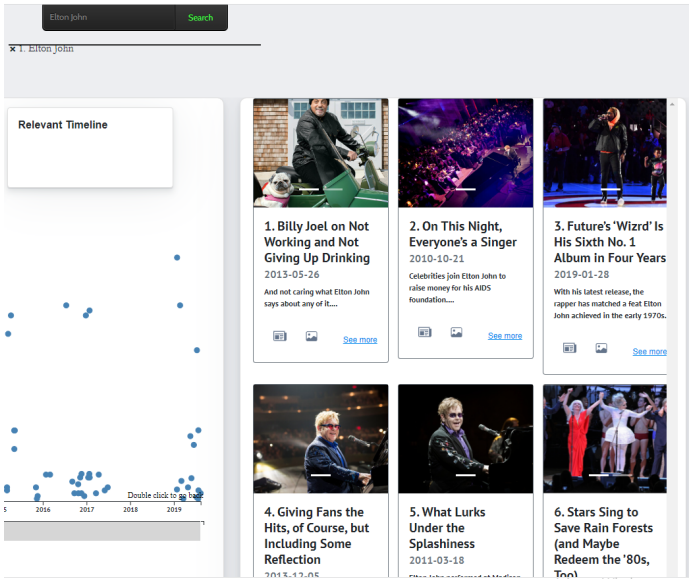


Figure 4.4: News feed resulting from the text query "Elton John".

4.2 System state

When a user has a specific topic in mind that he wishes to explore, the user might not obtain the desired results in the first search query, meaning that he might not be able to obtain the information that he wishes to gather at the firt try (first search). There is also the possibility that the user does not have a specific topic in mind but rather wishes to explore for a more general topic and, as he gathers information, refines consequent searches and eventually completes his exploring purpose. In both cases, it is likely that the user will need to make more than one search query to eventually be satisfied with the results. It is also highly likely that although a search query did not provide the desired results, it can be used to refine consequent searches, as although past searches did not provide the desired results, it is likely related to the actual desired results of the user. What this means is that the next search queries can benefit from taking into account past search queries when it comes to decide which news articles should be returned as well as their relevance. This can be seen as the system being "aware" of the user's past searches. This system "awareness" can be beneficial for the purposed work and it was with this in mind that it was implemented.

As described in section 3.4 the State Aware service classifies news through a score between 0 and 1. This score indicates the degree of relevance of each news article. The State Aware service takes into account previous searches when deciding a news score in hopes that it will provide better results as means of more relevant news articles to the user. For example, Figure 4.5 shows the 6 most relevant news for the search word "Putin". Figure 4.6 shows the 6 most relevant news articles for the search words "Ukraine", "Invasion" and "Putin". As it can be seen, 5 of the news presented in Figure 4.5 are presented in Figure 4.6, although in a different order, meaning that the State Aware

service decided that due to past searches, the relevance of a news article of a specific search word changed in hopes of meeting the user's expectations.

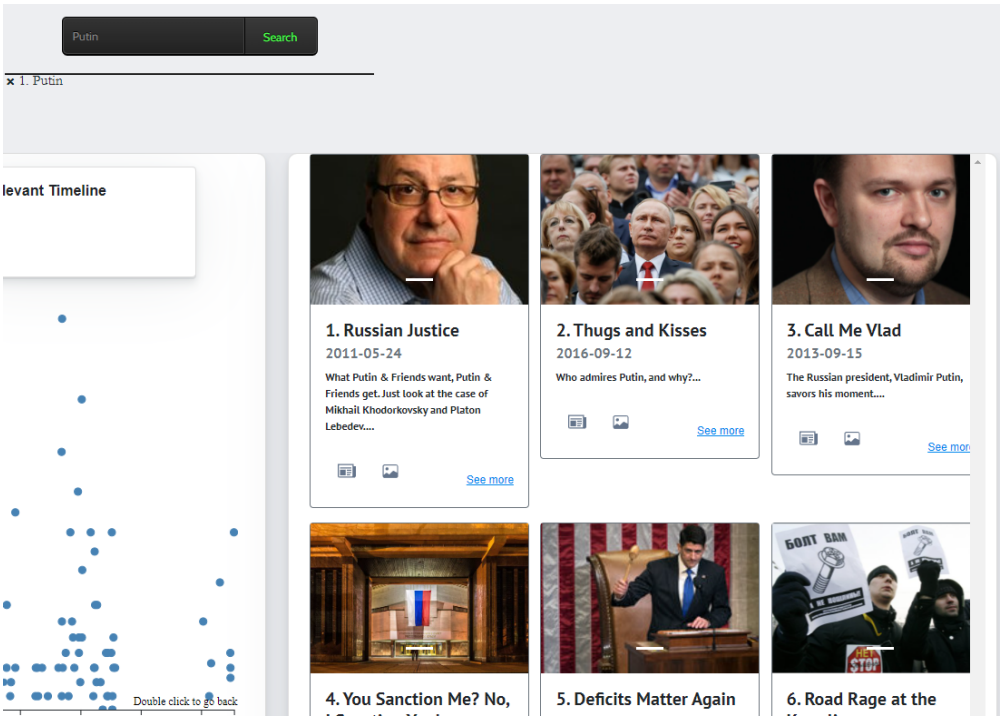


Figure 4.5: News feed resulting from the text query "Putin".

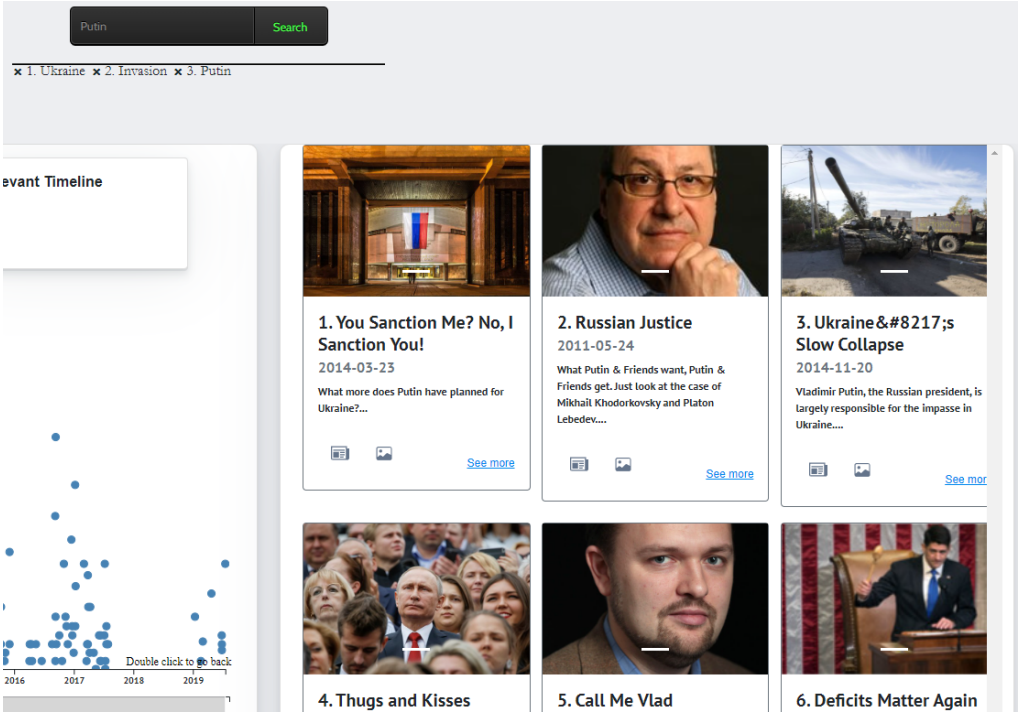


Figure 4.6: News feed resulting from the text queries "Ukraine" followed by "Invasion" and "Putin".

4.2.1 System State-Search

The State Aware service system was designed to be stateless, meaning that this service does not keep track of past interactions. Past interactions are stored in the user's side (web application), as this is necessary for caching reasons that will be presented in Section 4.9.

In order for the State Aware service to take into account previous searches and be able to sort the current news by relevance, it needs to receive two arguments: An array of past interactions and the current interaction. An interaction is composed of two fields, the type of the operation which indicates if the executed query operation was textual or visual and its results, which is an array containing the IDs of the retrieved news. For example, if the user made a search query using the search bar, the interaction object would be presented as the following: new-interaction: {Operation: "text", Results: [newsID_1, newsID_5, newsID_27...]}

Figure 4.7 explains the system state workflow and its steps. The system steps will be referenced during the following workflow explanation.

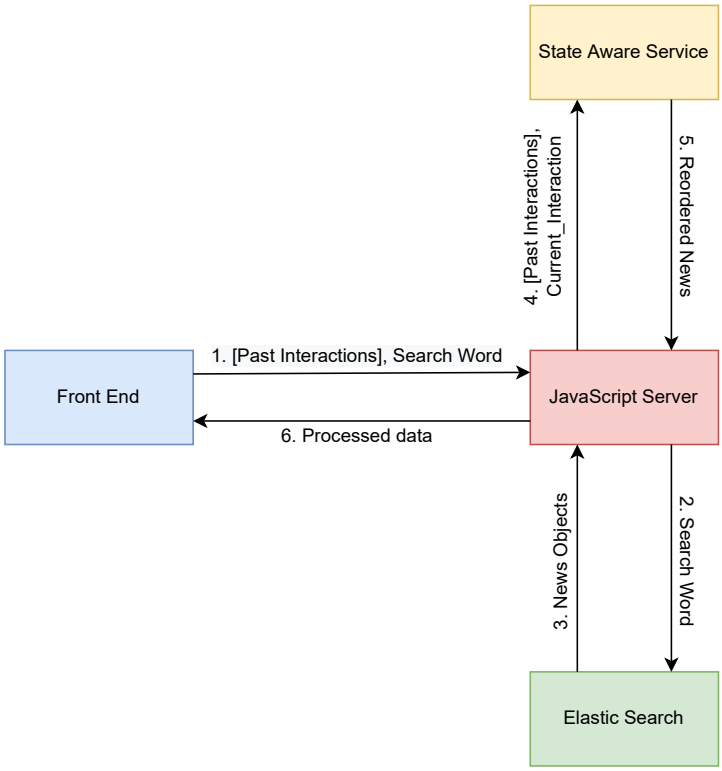


Figure 4.7: Image of the system state workflow

Whenever a user makes a search, the user side sends all the past interactions to the JavaScript server alongside the searched word (step 1.). The JavaScript server then sends a request to the ElastiSearch, passing the searched word as an argument and retrieving the news content (step 2.). After the JavaScript server receives all the news content from the ElasticSearch component (step 3.), the JavaScript server extracts the newsIDs and creates a "current interaction" object using the extracted newsIDs and the operation type.

Afterwards, the JavaScript server makes another request to the State Aware service that receives the current interaction alongside an array of all past interactions type and results (step 4.). The State Aware service then reorders the news according to the current and past interactions and sends the news ordered by score (step 5.). This way, news articles are already sent from most relevant to least relevant.

The implementation of the State Aware service was implemented by Cláudio Bartolomeu in another thesis parallel to this one, the connection with the different components of the system was made in this dissertation.

4.3 Similar News Retrieval

Since one of the main goals of the system is to provide users with ways to retrieve news articles that meet their desire, it can be beneficial to the user to have features that allow to retrieve news articles similar to one of the user's interest. By implementing such features, the user is provided with the ability to retrieve news articles similar to the ones of his interest.

When developing news articles, content creators can also come across the need to find images to place in their developing news articles. Generally, news articles images have a connection to the content of the news articles, either to offer visual context and use the image to explain a situation or simply to enrich their developing news article with images that are in line with their developing news content. In both cases, there is a need to have news articles retrieved through either the content of the news article or simply by the visual content of the news article image.

The aforementioned abilities were implemented to provide the user with the capability to retrieve similar news articles based on another news article content or images. The implementation relied on the use of buttons which are present in each news article. Figure 4.8 shows both news articles buttons. News articles are composed by the news text and its images. The system is capable of identifying and retrieve news articles that are similar to another based on their content, making it a multimodal search. A multimodal search uses the ability of the system to retrieve similar news while taking into account both the text and the visual aspect of the image of a specific news article, meaning that in theory the returned news articles will be similar in regards to the text and visual content of the specific news article.

The system is also able to identify news articles that are similar solely based on the visual aspects of a specific news article, images making it an image-based search. Contrarily to the multimodal search, if a user decides to retrieve news articles solely based on another news article image, then the system retrieves news articles that have images similar to the one of interest without taking into account any aspect of its text content, meaning that the retrieved news articles can be totally unrelated regarding their text.

As the multimodal search was one of the explored features. If the user finds a news article interesting and thinks that retrieving similar news articles to that one can be

beneficial, he can do so by clicking on the button presented in Subfigure 4.8(a). This will retrieve similar news articles based on the whole content of the original news article. By clicking on the button presented in the Subfigure 4.8(b), the user will retrieve similar news articles taking only into account the visual aspect of the images of the news article.

(a) Multimodal search icon

(b) Image-based search icon

Figure 4.8: Similar news icons.

4.3.1 Similar News Articles (Multimodal Search)

In case a user is exploring the news feed and finds some news article relevant, he might want to find similar news articles to that news article. By clicking on the button presented in Figure 4.8(a), the system will retrieve the top 100 similar news to the clicked one. The news feed will then show these 100 news articles and all visualizations will be updated with the current shown news. It is important to mention that when retrieving the top 100 news articles, the ranking order is decided by taking into account both the text of the original news article and only the visual content of the current image shown at the time of the click of the icon when choosing the top 100 similar news articles. For example, if the image shown in the news article at the time of the click of the button is the first one, then the visual content taken into consideration is of that image. If the second image is shown, then visual content of that second image is the one taken into account and so on. By considering both the text of the original news article as well as the current image shown at the time of the click when ranking the news, it insures that the returned news contents are as similar as possible to the original news.

Figure 4.9 shows the news feed after searching for "Roger Federer Wimbledon". All the news articles displayed on the news feed are related to the tennis star Roger Federer and the Wimbledon tennis championship. Figure 4.10 shows the resulting news feed after clicking on the "Multimodal search button" button of the first article in the feed of the Figure 4.9.

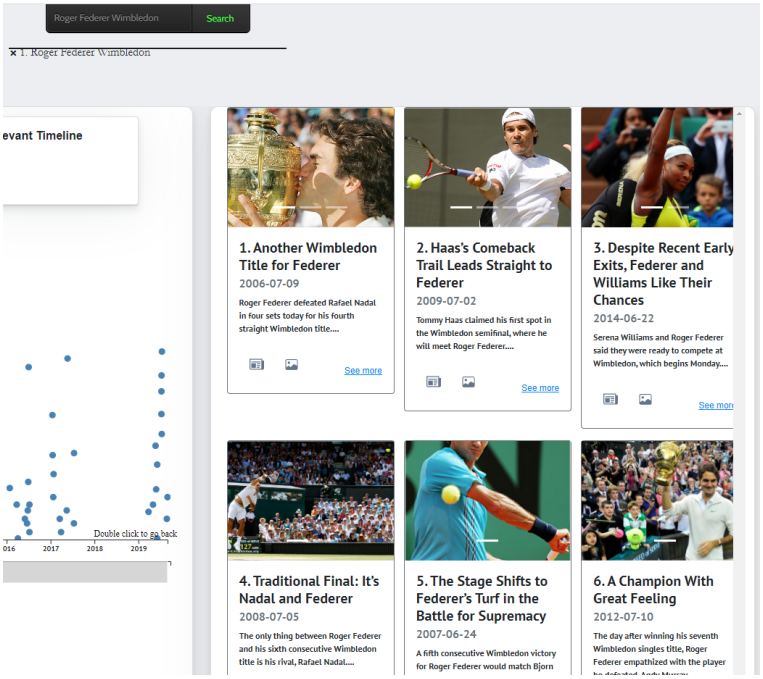


Figure 4.9: News feed resulting from the text query "Roger Federer Wimbledon".

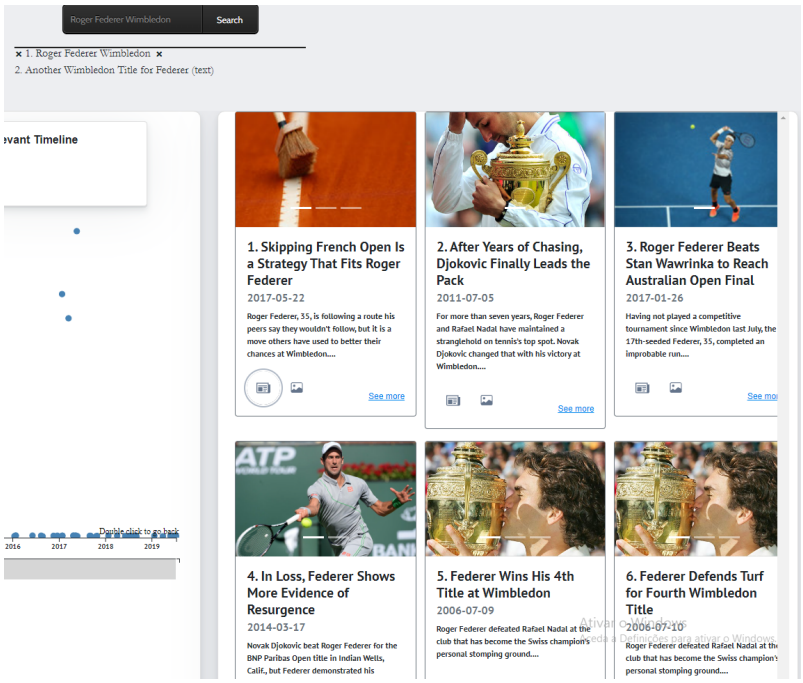


Figure 4.10: News feed resulting from clicking on the multimodal search button.

4.3.2 Similar News images (Image-Based Search)

As previously mentioned, news articles also have a button (Figure 4.8(b)) that allows users to perform an image-based search query returning news articles that have similar

images to the news article image of the clicked icon. Similarly to the one described before but only taking into account the visual aspect of the news image.

This feature was implemented considering that sometimes content creators might not be interested in finding or exploring news articles, but might simply have the desire to find certain images to include in their developing news articles. Due to this fact, content creators might be interested in finding a picture that best represent the content of the news articles that they are developing. For example, if a content creator is writing an article about political negotiations between world leaders (e.g.,), the Chinese prime minister Xi Jinping and the former American United States president Donald Trump, then the content creator might be interested in finding images that picture a political event between the two. The user can, for example, firstly search for "Xi Jinping, Trump" to obtain news articles between the two (Figure 4.11) and find a relevant image. Afterwards, the user can click on the Image-based search button of a news article that has the type of image that the user is looking for in order to obtain similar images to the desired one. Figure 4.12 shows the result of the news feed after clicking on the Image-based search button of the fourth news article presented in Figure 4.11 which depicts a political event between the two intrevenientes of interest. As it can be seen, the images in the resulting news feed are similar to the original image of interest. Most of the images have the same characteristics, such as two country flags behind political representatives, the images also clearly depict a political event containing only a few people and 3 out of the 6 presented images contain both the president Xi Jinping and the former president Donald Trump. However, although it is clear that the images are somehow related, some of the news articles do not contain either Trump or Xi Jinping in their images, which was the intended purpose. In that regard there is room for improvement.

4.3.3 Similar News Implementation

Although the function of the multimodal and image-based search buttons is different, in reality their implementation is practically the same, differing only in one of its workflow steps. Figure 4.13 explains the "similar news" workflow and its steps. The workflow steps will be referenced during the following workflow explanation.

When one of the buttons of a news article is clicked, the news ID and the image order of the news article is sent to the JavaScript server alongside an array of past interactions (step 1.). For example, if a user clicked on one of the buttons and the ID of that news article is 501 and the current image shown at the time of the click of the button was the second one, then the NewsID_ImageOrder would be 501_2. After the step 1 is completed, the JavaScript server then redirects the newsID_imageOrder to the Python server (step 2.). As previously described in Section 3.4, the Python server makes use of two indexes, a visual index and a multimodal index. These indexes group embeddings in a way that embedding similarity searches can be done. The visual index allows for similarity searches based on images, while the multimodal index allows for similarity searches based on the content

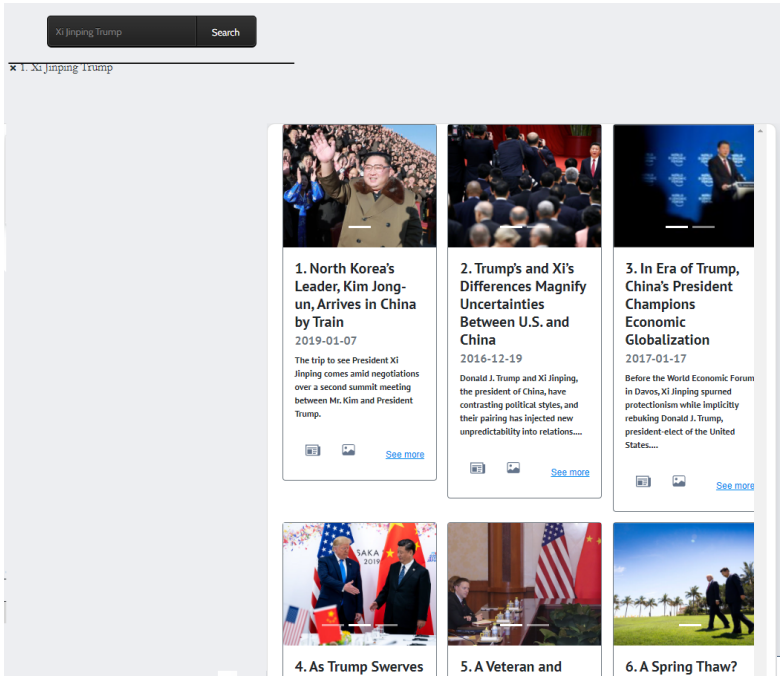


Figure 4.11: News feed resulting from the text query "Xi Jinping Trump".

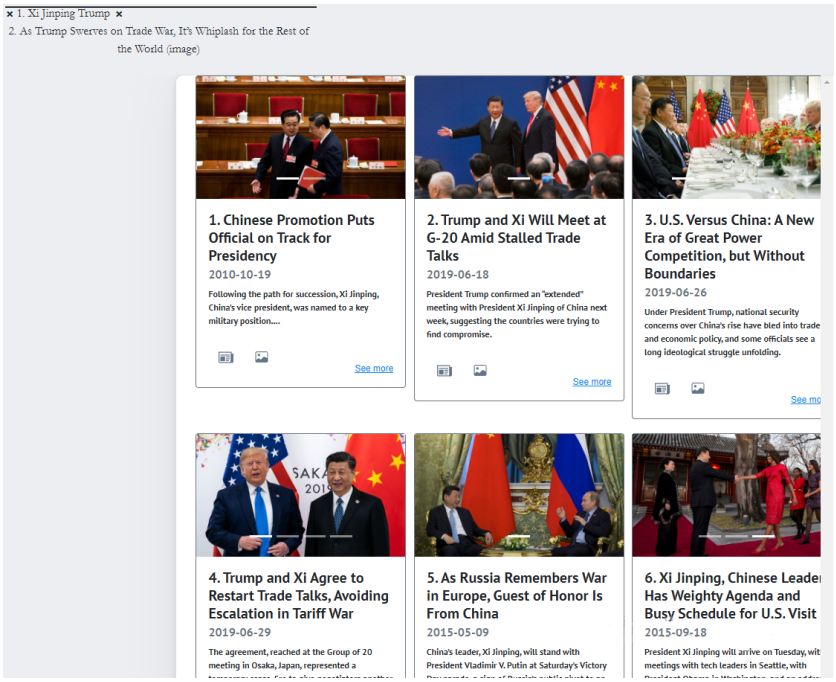


Figure 4.12: News feed resulting from clicking on the similar news icon.

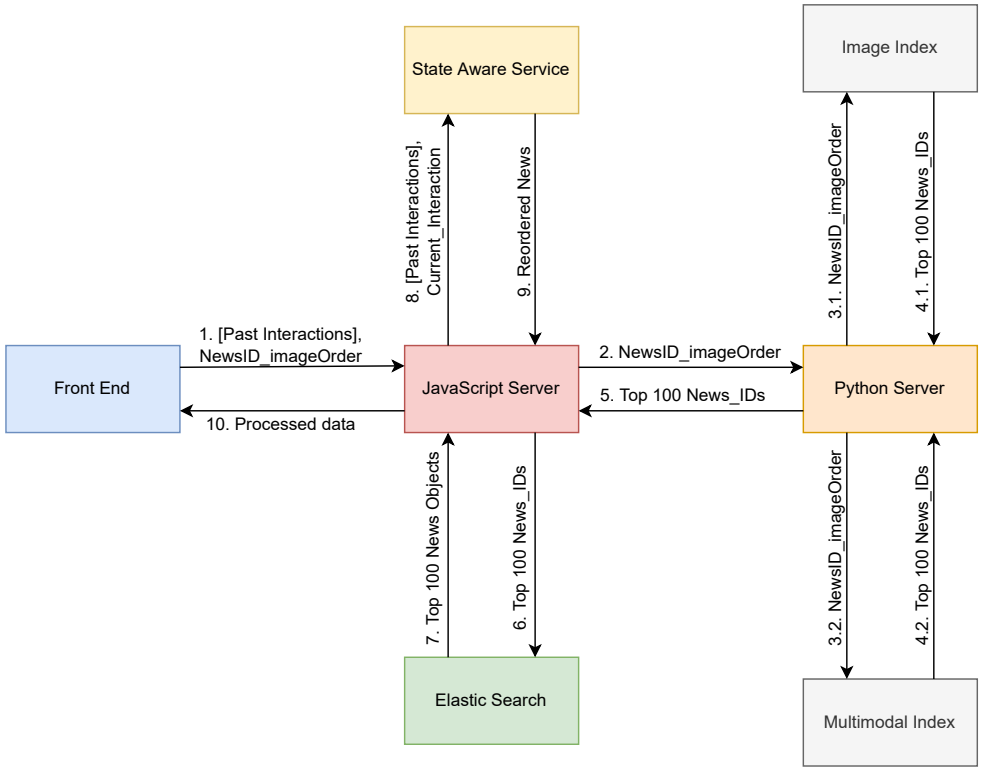


Figure 4.13: News feed resulting from clicking on the similar news icon.

of a news article. If the user clicked on the image-based search button, then the Python server extracts the embeddings of the NewsID_image using a stored visual index (step 3.1). The extracted embeddings visually describes the image through a series of arrays. It can be seen as transforming the visual representation of an image to a series of arrays of values. In case the clicked button was the multimodal one, then it uses the stored multimodal index (step 3.2), which can be seen as transforming the visual and textual representation of a news article to an array of values. This is the only part of the code that differentiates between the two buttons. Afterwards, the extracted embeddings values are used to query the corresponding index in order to retrieve the 100 most similar newsIDs (step 4.1 or 4.2). This is done by comparing the extracted embeddings to other news articles embeddings. The 100 NewsID's are then passed back to the JavaScript server (step 5.). The JavaScript server then sends a request to the Elasticsearch with the list of NewsIDs to be retrieved (step 6.). The Elasticsearch answers with the content of the 100 NewsIDs requested (step 7.). Finally, the JavaScript server creates a "current interaction" object using the top 100 news IDs and makes another request to the State Aware service that receives the current interaction alongside an array of all past interactions type and results (step 8.), which orders the news according to previous made searches and sends it back to the JavaScript server (step 9.). The ordered news content are then sent back to the web platform (step 10.).

4.4 Time Analysis Visualization

Time-based exploration techniques place a relevant focus on document dates and their distribution throughout time. One advantage of this approach is that it is possible to spot sudden bursts of news articles at a certain point in time, which might mean that a "relevant" event might have occurred, although this is not necessarily true in all cases. It is also likely that these sudden bursts can catch the user's attention. These sudden bursts of news can also provide users benefits if the user is in exploration mode, learning more about a topic. If a certain topic search provides many burst spots throughout time, then it is possible that other burst spots represent an event or its unfolding throughout time regarding that specific topic.

The Time Analysis visualization was implemented in order to take advantage of the benefits of obtaining a distribution of news over time, such as the ones previously described. One of the goals when implementing this visualization was to also provide users a way to refine his search through this view by selecting, for example, only news articles corresponding to a certain time frame.

4.4.1 Time Analysis User Interface

As the name indicates, the Time Analysis visualization serves the purpose of analysing the spread of news of a certain search topic throughout time. When a user makes a search request and the system returns a set of news, that same set of news is represented in the Time Analysis visualization. Figure 4.14 shows the resulting Time Analysis visualization after searching for the word "Ukraine". Every blue dot in the visualization represents one of the returned news article. In the figure, there are 200 blue dots. In the X axis, news are positioned according to the date of the news article, which in the case of the figure, ranges from 2006 to just after 2019. In the Y axis news are positioned based on the score that the system gave to that specific news article, the score always ranges from 0 to 1 and the higher the score the more relevant a news article is.

If the user wants to explore a specific news displayed on the visualization, the user can simply hover the mouse over one of the blue dots. The corresponding dot will grow larger and the corresponding news article will then be displayed in front of the visualization as shown in Figure 4.15. If the user wants to lock a specific news to further explore that specific news article, such as, clicking on the news article icons to return similar news, access the full news story, views its images, etc, he can do so by clicking on the corresponding blue dot, the displayed news article will turn static until the user hovers another blue dot.

One of the objectives of this visualization is to offer the user the ability to understand which time periods were the most relevant for a certain topic, as this can situate the user in a time frame as he explores a certain topic. By having a visualization that displays the density of news throughout time, it can also draw the attention of the user to certain

time frames if a specific time frame is much more dense news article wise than other time frames. For example, Figure 4.14 shows that a lot of the 200 shown news (represented by the blue dots) are concentrated between the 2014 - 2015 area, which corresponds to the time Russia invaded Ukraine. This event was surely a very mediatic event at the time and certainly one that news creators might be interested in exploring if they want to develop news articles about the Ukraine invasion. Also, if a user is in exploration mode, learning more about a topic, it also draws the attention of the user who can then gain more insight about that specific "relevant" time period.

Lastly, if users are interested in exploring specific time lines, they can do so by selecting a time range, placing the mouse cursor in the desired start by clicking on the visualization and dragging it to the desired end date. A slider will be made visible indicating the range of the selected time line. Figure 4.16 shows this interaction. After the user selects the desired time line, the time analysis visualization updates and only shows the selected time range and blue dots that fall within that range. Figure 4.17 shows the resulting interaction. Similarly, the news feed only shows the news that are within the date range. This allows the user to filter the amount of news by date and narrow the amount of information shown in the visualization. The date on the cards located at the top of the figure also update their data according to the number of news items selected. In this case, the amount of news selected was 98, which corresponds to 49% of the news initially returned in the first search. The precise selected timeline is also shown. The process of selecting a specific time range through the brushing tool can be done as many times as the user wants, with each brushing narrowing the time range and the amount of news shown, providing the user with the capability of continued filtering. In case the user wants to go back to a previous state, the user can double click on the time analysis visualization undoing the last filtering. The user can undo a filtering operation as many times as he filtered, with each double click undoing a previous brushing action.

4.4.2 Time Analysis Implementation

Regarding the time analysis implementation, this visualization only requires the content of the articles, its dates and score in order for the visualization to be loaded. This visualization does not need to transform or process any kind of data. After the user makes a search query and the corresponding news articles are returned to the client, all the data needed for the time analysis visualization to be displayed is present.

The Time Analysis visualization was implemented with the help of the D3.js library and other react components

4.5 Embeddings Visualization

As previously explained in Section 2.3, embeddings models map high-dimensional objects into lower-dimensional continuous vector spaces such that the vectors of related objects

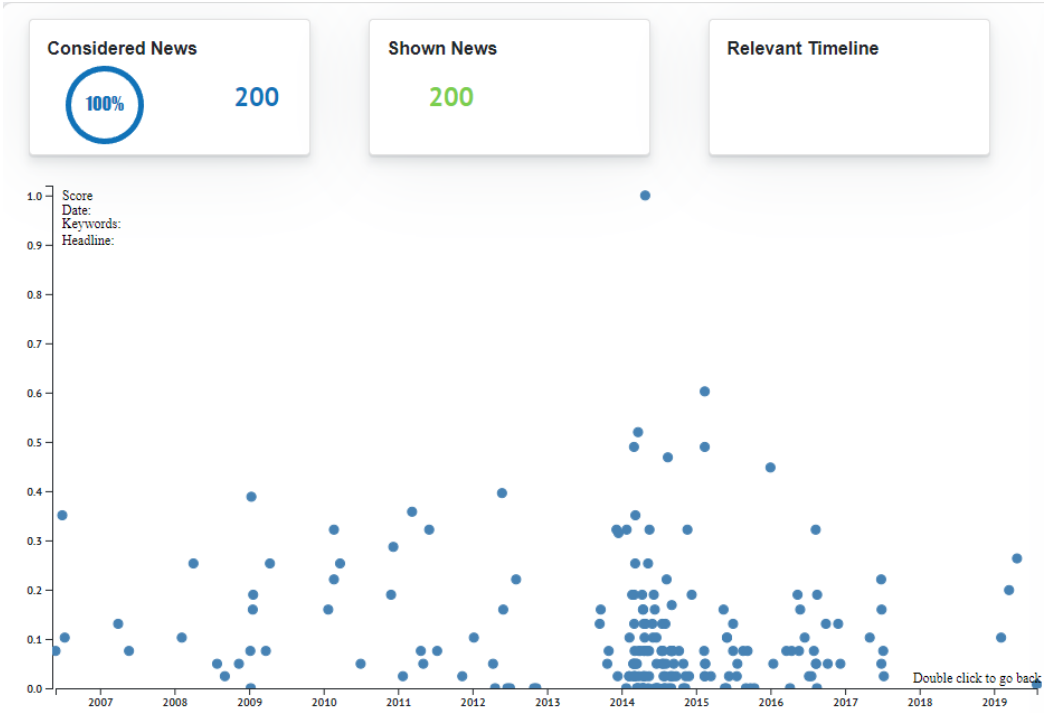


Figure 4.14: Time analysis visualization after searching for "Ukraine".

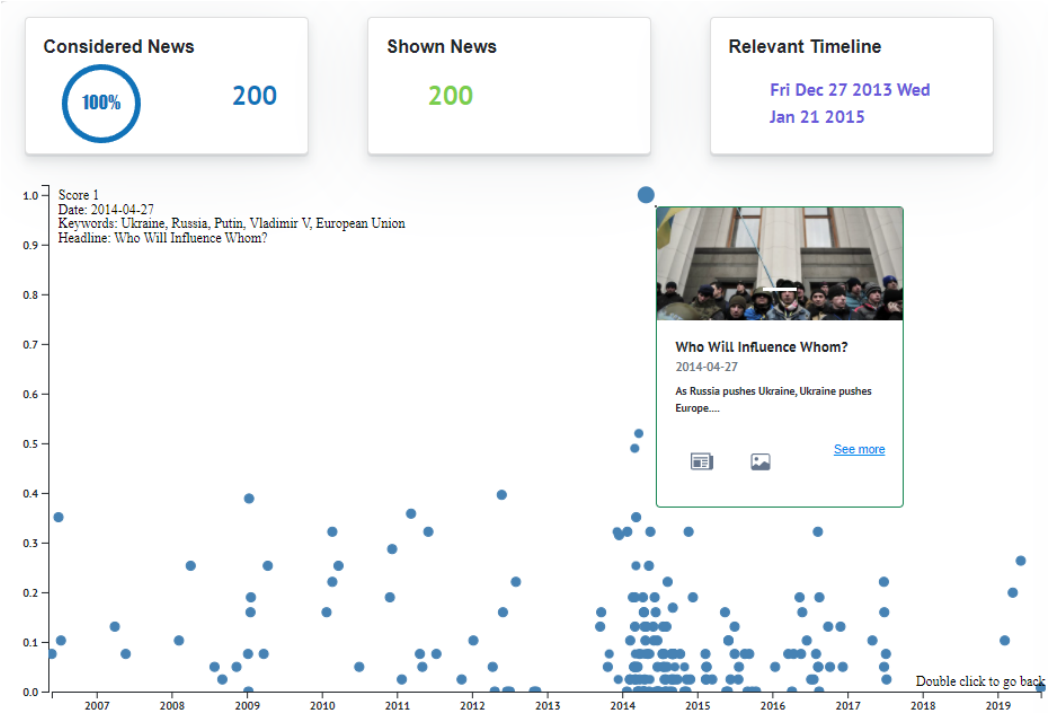


Figure 4.15: Time analysis visualization showing a news article after hovering its respective blue dot.

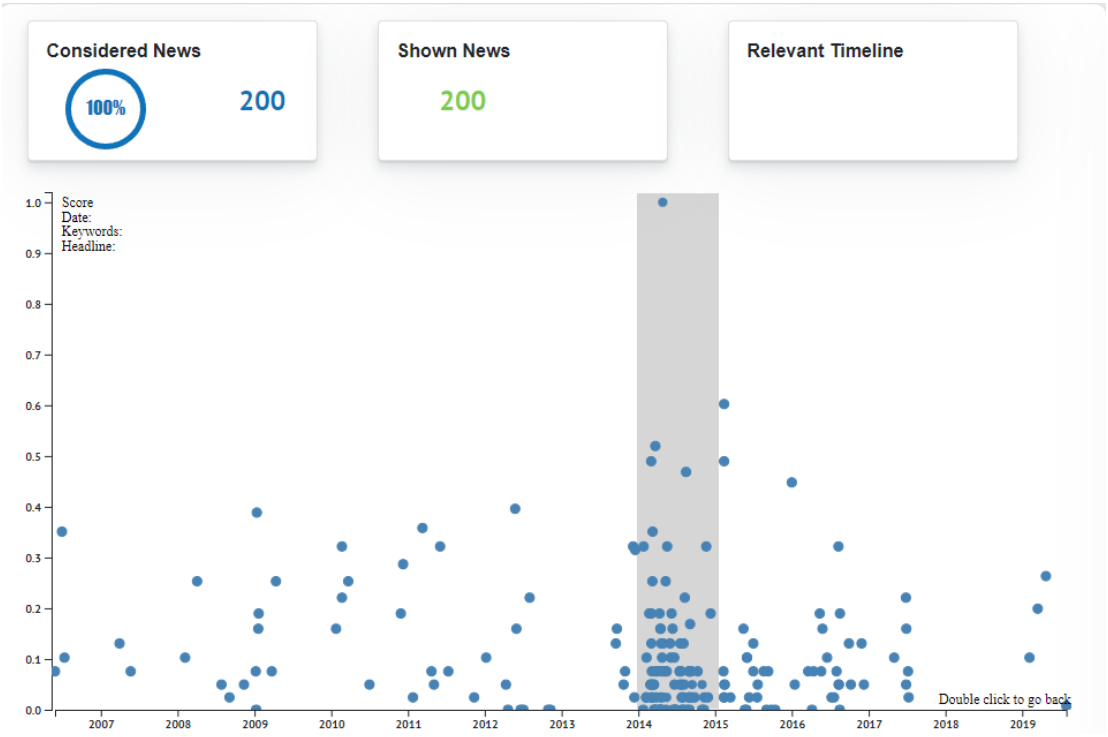


Figure 4.16: Image showing the slider tool of the time analysis visualization.

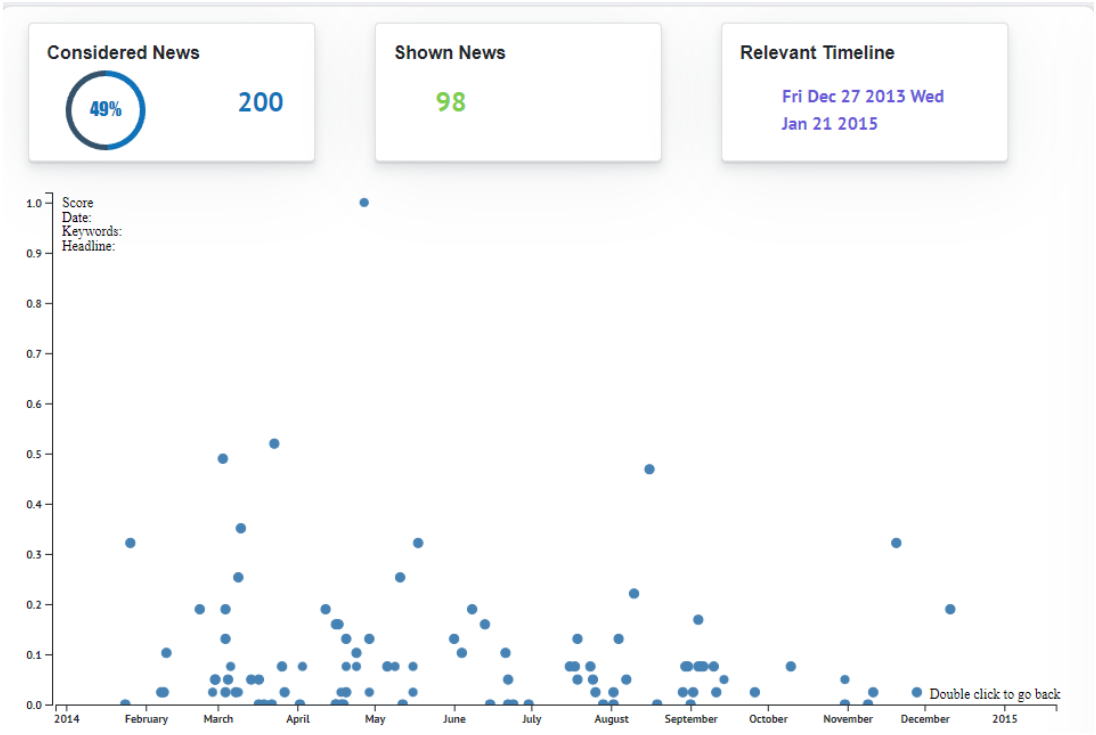


Figure 4.17: Time analysis visualization after filtering the amount of news through the use of the brushing tool.

are located close together. Embeddings vectors are sets of values that can represent something abstract like the visual content translated to a set of values. This means that the visual content of an image can be translated in a set of values.

Generally embeddings visualizations, like many visualizations, rely on a 2D space representation. Although perhaps implementing a 2D visualisation is a safer option, it was decided to use a visualisation of a 3D space as a mode of representation, as it might go outside the norm and possibly create curiosity of exploration in the user.

The goal of the Embeddings visualization is to explore news articles in a 3D space. This implies that news articles have to be positioned in the 3D space according to some coordinates. One of the goals of this visualization is to have news articles positioned according to their content. This implies that news articles that are similar in their content would be placed closer than those that are not. In order to achieve this, the coordinates of each news article are calculated in such manner that depends on the contents of the news. When defining the news articles coordinates, it takes in consideration both the news articles images and text. This means that the closer the news articles are to each other the more similar they are, both textually and visually.

4.5.1 Embeddings User Interface

After the user performs a search and receives the corresponding news, the Embeddings visualization is displayed. Figure 4.18 depicts the resulting embeddings visualization for the searched word "Trump". The figure shows five spheres. Each sphere has a collection of news articles associated with it, meaning that the spheres contain several news articles inside them. There is a correlation between the sphere size and the amount of news articles that it contains, generally, the bigger the sphere the more news articles it contains. Spheres are also positioned according to the content of the news articles that belong to that sphere. For this reason, the closer the spheres, the more similar their news articles are. At the top of each sphere lays the most frequent keyword of the news articles that belong to that specific sphere. If a sphere is hovered, then the four most frequent keywords of all the news articles inside the sphere become visible. Figure 4.19 shows the previously described interaction. The most frequent keywords of the hovered sphere are: "Trump, Donald J", "United States Politics And Governament", "Presidential Election of 2016" and "Presidential Election of 2020". This means that the sphere contains news articles that address those subjects.

Since this visualization places the spheres according to given coordinates, it is possible that some spheres fall outside the embeddings visualization screen. Some fall so outside the visualization screen that it might be hard for the user to find it without help. Due to this fact, at the top of the visualization screen lays a smaller sphere with arrows that point and follow each created sphere (Figure 4.18).

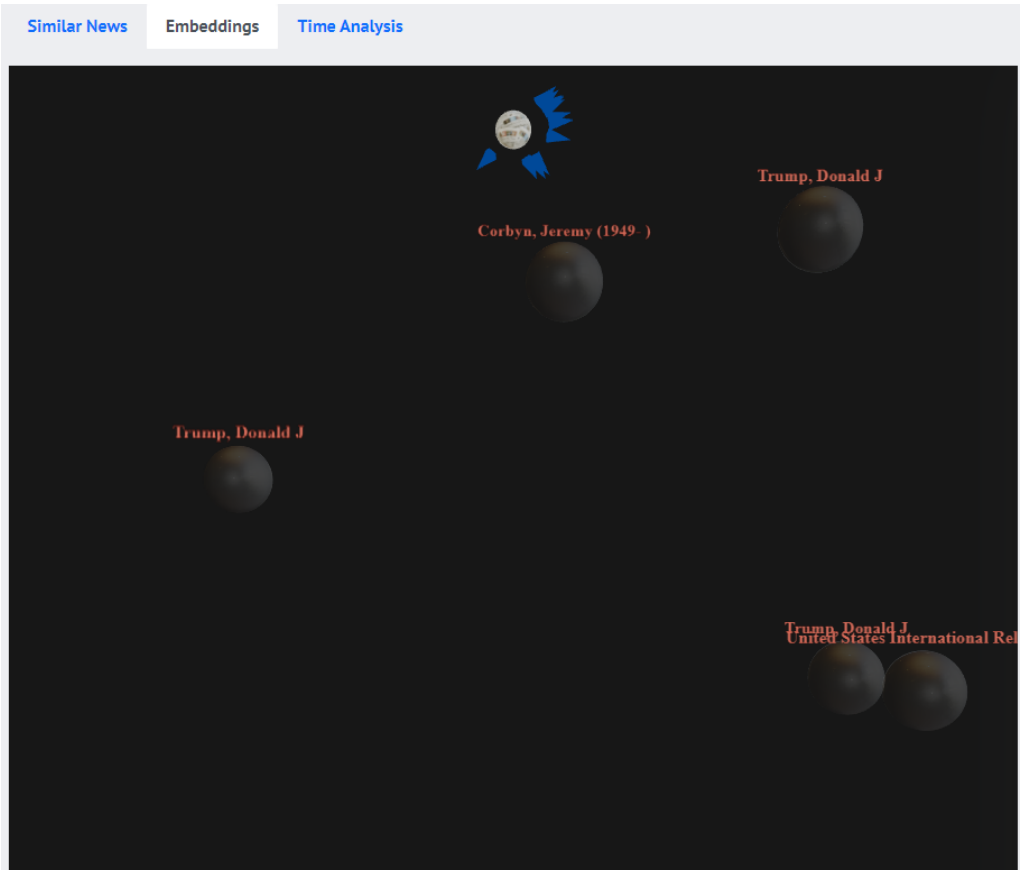


Figure 4.18: Embeddings visualization after searching for "Trump"

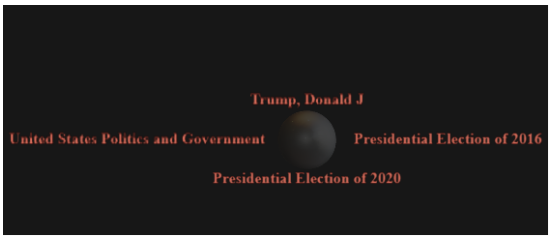


Figure 4.19: Hovered sphere with the respective keywords visible

The user is able to navigate through the visualization screen by dragging it with the mouse. If the user wants to explore the contents of a sphere because he found the keywords to be in consonance with what he seeks or simply by wanting to, the user can click on the sphere he wishes to explore. This will trigger an animation that zooms in the center of the sphere and at the same time loads the pictures of the news articles belonging to the clicked sphere. All the other spheres remain invisible until the user zooms out of the current sphere that he is exploring. Figure 4.20 shows the resulting interaction. As it can be seen, the figure displays the images of the news articles and, as previously described, the position of each news article depends on its content, placing similar news articles closer to each other. The user can now explore the news article

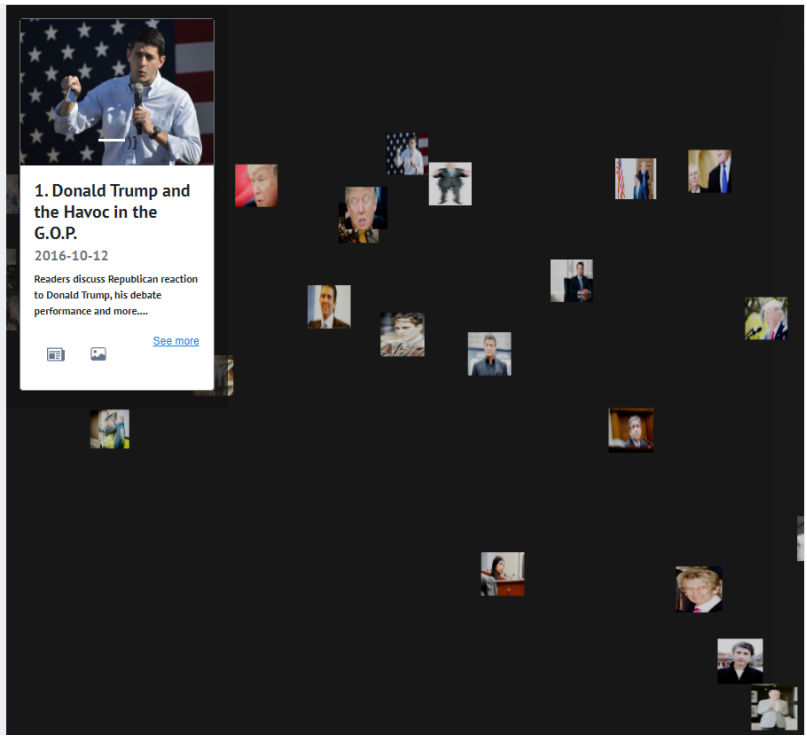


Figure 4.20: Resulting interaction by clicking in a sphere. Images are placed according to their content

cluster, by dragging it with the mouse, to position the screen camera where he wishes. The user is also allowed to zoom in to take a closer look at the images. The news article's images have three degrees of quality: low, medium and high. The closer the zoom the better quality of the images. This feature was introduced to reduce the amount of memory needed when showing the images, as many high quality images loaded at the same time used too much memory, making the visualization's interaction slow.

Lastly, when the news articles are shown and a user hovers a news article, that specific news article is shown at the left side of the visualization, this can also be seen in Figure 4.20. This is important because the visualization only shows the images of the news articles, and this way, the user can observe which news article corresponds to the image. All the features of the news articles previously described are available to the user.

4.5.2 Embeddings Implementation And Workflow

Figure 4.21 represents the workflow of the embeddings visualization.

When the web application receives a request sent on the search bar (step 1.), it makes another request to the ElasticSearch server using the same word(s) searched by the user (step 2.). The ElasticSearch server then returns a response that contains the top 200 news articles content ordered by a score that best matched the searched word(s) (step 3.). These news articles objects contain every information needed for the purpose of creating the embeddings visualization. The news IDs are then extracted and, for each image that

a specific news has, the order of the image is appended to it's ID and added to a list. For example, if two news articles with "id1" and "id2" both have two images, the list would look like: [id1_0, id1_1, id2_0, id2_1] and so on. After this process is completed, the JavaScript server makes a request to the Python server and sends the list previously described (step 4.). The Python server receives the list and, for each element in the list, extracts it's embeddings and stores them in an embeddings list. News embeddings are vectors representing news pieces in a vector space of "x" dimensions, where "x" is hundreds of dimensions in our case. Each element embeddings generally contains several hundreds of dimensions.

Due to the fact that we are dealing with hundreds of dimensions, there is a necessity to reduce these dimensions to at most three dimensions so they can then be visualized in a 3D space. For this reason, the dimensionality reduction algorithm UMAP is then applied onto the embeddings list reducing the embeddings dimensionality from hundreds of dimensions to three dimensions. These dimensions will then be used as coordinates in the visualization. Afterwards, the density based clustering algorithm DBSCAN is applied to the three dimensions list creating clusters based on the more dense regions, which are regions where there are groups of news with similar coordinates and therefore closer to each other. Each news article will either belong to a cluster or be labeled as an outlier. Each cluster contains a group of news that are similar to each other.

It is important to mention that the DBSCAN density clustering algorithm uses parameters to establish how these clusters are created. One of those parameters is the *"eps"* parameter, which defines the maximum distance between two samples for one to be considered as in the neighborhood of the other. The *eps* paramenter is seen as the most

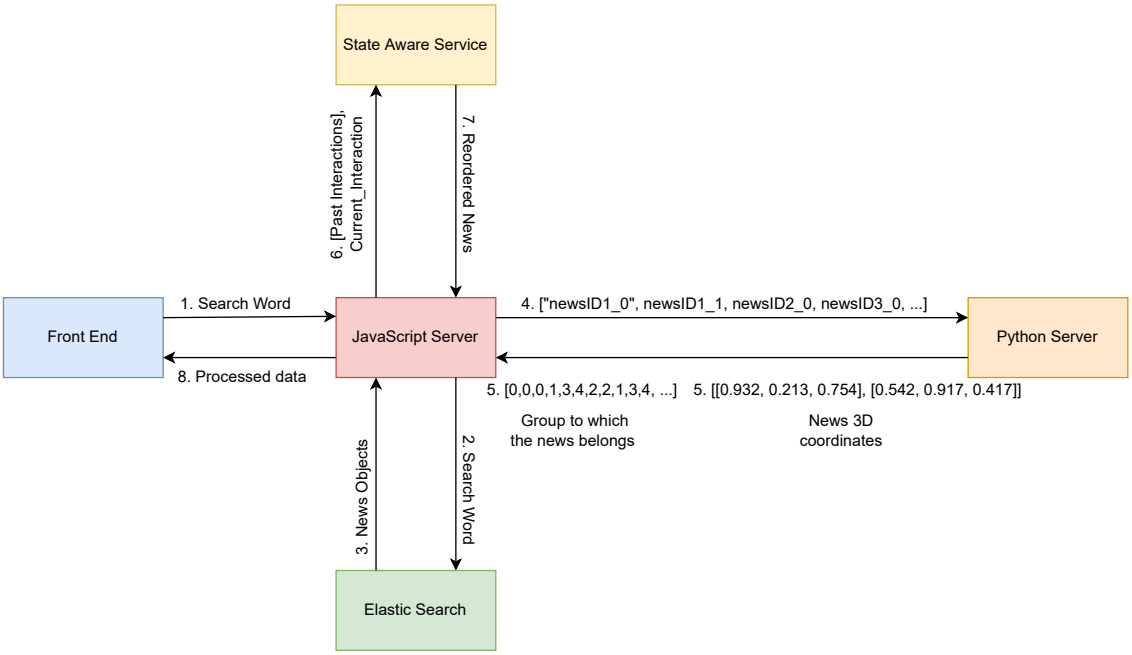


Figure 4.21: Embeddings visualization workflow

important DBSCAN parameter to choose appropriately for the data [48], this parameter is responsible for determining the amount of spheres created in the visualization. The *eps* parameter is crucial because a not so optimal value of this parameter might result in a not so optimal result. For example, if the *eps* value is too high then the algorithm will consider every point as within the same cluster, creating only one big cluster and placing on the same cluster news articles that are not similar. On the contrary, if the *eps* value is too low, the reverse happens, considering every point as an outlier. The optimal value generally lays somewhere in between.

Nadia Rahmah et al [62] conducted a study to try to automatically find the best *eps* value for a dataset. In the paper, the author calculates the average distance between each point and its k-nearest neighbors, where k is equal to a pre-selected number of points. The average k-distances are then plotted in ascending order on a k-distance graph. The optimal value for the *eps* value will correspond to the point of maximum curvature of that graph. Figure 4.22 presents the pseudo code to find the best *eps* value. Figure 4.23 shows an example of an expected curvature alongside the best *eps* value for a certain dataset.

Algorithm 1 The pseudo code of the proposed technique DMDBSCAN to find suitable Epsi for each level of density in data set	
Purpose	*To find suitable values of Eps*
Input	*Data set of size n*
Output	*Eps for each varied density*
Procedure	*1 for i* *2 for j = 1 to n* *3 d(i,j) ← find distance (x_i, x_j)* *4 find minimum values of distances to nearest 3* *5 end for* *6 end for* *7 sort distances ascending and plot to find each value* *8 Eps corresponds to critical change in curves*

Figure 4.22: Pseudo code in order to the best eps value [62]

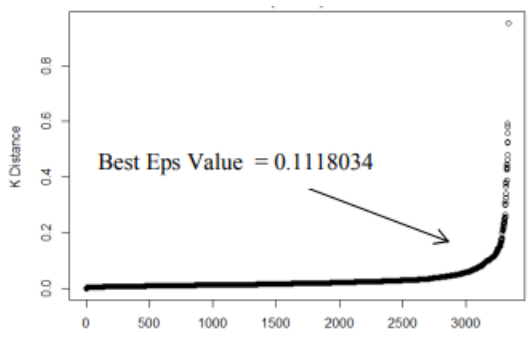
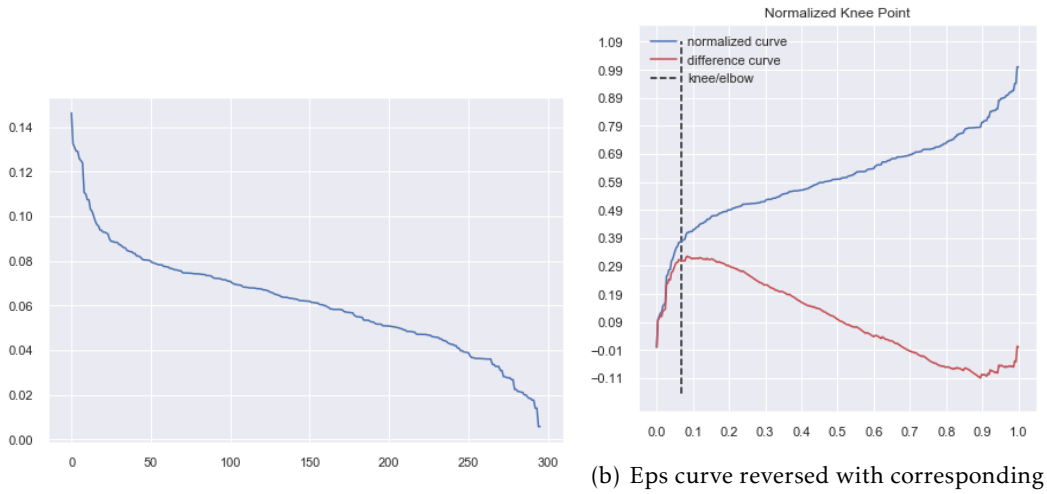


Figure 4.23: Eps curve [62]

In our case, initially, the eps value was constant. Although a constant value of eps performed reasonably well in some searches, as it formed clusters with a reasonable degree of similarity between news articles, it also performed poorly on other searches, where it would not create any cluster or it would create only one cluster with all the

news articles inside. For this reason, the technique described before was adopted and the eps value was now automatically found for each search. The approach provided much better results as the clusters contained a higher percentage of articles that were actually similar to each other. Subfigure 4.24(a) shows the resulting curvature for the search word "Trump" where the average k-distances are plotted in ascending order on a k-distance graph, with k = 6. As it can be seen, the maximum curvature point lays somewhere in between X values of 0.07 to 0.12. To automatically find the best eps value, the graph is then reversed and the python library kneed [1] was used to find the maximum point of curvature, also known as knee point. Subfigure 4.24(b) displays the reversed graph with the maximum curvature point located with the dashed line, with the best eps value being equal to Y = 0.95.



(a) Eps curve

(b) Eps curve reversed with corresponding best eps

Figure 4.24: Eps curve for the word "Trump"

Another parameter used by the DBSCAN algorithm is the min-points [2], which is the number of samples in a neighborhood for a point to be considered as a core point. Sander et al [66] proposed to set the number of min-points to 2 * (dimension of the dataset) - 1. Since the data being worked on is three-dimensional, the number of min-points is set to 5.

After the clustering process is finalized, the Python server sends two lists to the JavaScript server: a list indicating which cluster each NewsID_ImageOrder element belongs to, and another list with the 3D coordinates of those same elements (step 5.). The JavaScript server then uses the received 3D coordinates and calculates the minimum and maximum value of each coordinate (x, y, z) for each cluster. Meaning that each cluster will have a set of minimum and maximum values for each coordinate. These values are

[1] https://kneed.readthedocs.io/en/stable/
[2] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

then stored and will be used to create a sphere around the images of each cluster. Finally, the JavaScript server creates a "current interaction" object using the 200 news IDs returned from step 3 and makes another request to the State Aware service that receives the current interaction alongside an array of all past interactions type and results (step 6.), which orders the news according to previous made searches and sends it back to the JavaScript server (step 7.). Lastly, the JavaScript server then sends the 200 reordered news articles, a list indicating which cluster each news article belongs to another list indicating the 3D coordinates of each news article in the visualization and lastly the maximum and minimum values of each coordinate (x, y, z) for cluster to the client (step 8.).

With the data received it is possible to create the visualization. The reason there is a need to have the minimum and maximum value of each coordinate (x, y, z) for each cluster is that for example, a cluster might contain 10 news articles, each news article contains their respective coordinates, for a sphere to be created around those 10 news articles it is necessary to have the bounding box encompassing all 10 articles, which is done by having the minimum and maximum for each coordinate encompassing all 10 articles. With these values, spheres can be created around each cluster of news articles with the center of each sphere and its radius being determined as the following for each sphere:

$$SphereCenterX = (MaxX + MinX)/2$$

$$SphereCenterY = (MaxY + MinY)/2$$

$$SphereCenterX = (MaxZ + MinZ)/2$$

$$SphereRadius = \sqrt{(MaxX - MinX)^2 + (MaxX - MinX)^2 + (MaxX - MinX)^2}$$

All the spheres created in the Embeddings visualization used the previous mathematical equations.

The implementation of the embeddings visualization was created using solely the basic elements of Three.js library without the use of plugins or external animations.

4.6 Similar News Visualization

When a user is exploring news on the news feed and finds a certain news article interesting, the user might want to find similar news to that one as this can go to the encounter of the user's search desires. It is also possible, however, that similar news might be on the path of the user search desires but not exactly what he is searching for, in hopes to consecutively continue on more relevant paths, the user might choose to return similar news of the previous similar news, and so on, to eventually end in a state where he is satisfied with the results.

Providing a visualization that supports the previously described observation is the goal of the Similar News visualization. The Similar News visualization uses a graph like visualization since the data and exploration of this visualization follows a tree like structure.

4.6.1 Similar News User Interface

If a user, while exploring a topic, encounters an interesting news article that grabs his attention, the user might be interested in finding similar news to that one. If that is the case, by clicking on the image of the news that he wishes to find similar news, it will put the news into focus and the tab "Similar News" will display a visualization that is meant to show the user similar news to the one that was put into focus. Figure 4.25 shows the result of the previously described interaction. On the right side of the figure the news feed is shown alongside the focused news article, which corresponds to the news article image that the user clicked. On the left side of the figure, the "Similar News" visualization is displayed. The visualization starts with a green node on the center which represents the focused news article. The text in the centre of the news item corresponds to the keyword that best describes the article. The corresponding focused news article is also displayed on the left side of the visualization.

The user can now start to interact with the green node. By hovering on top of the node the circle around the node expands, meaning that specific node contains similar news and therefore can be expanded (Figure 4.26). Saying that a node can be expanded

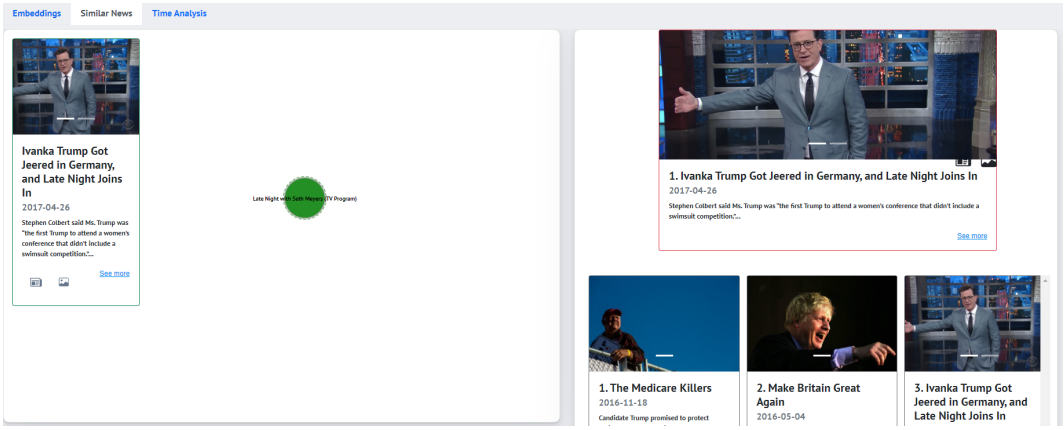


Figure 4.25: Similar News visualization (left) and the news feed (right)

Figure 4.26: An hovered node.

means that if the user decides to click on it, then five other nodes will appear in a star shape around the clicked node. If an already expanded node is clicked, it goes back to its original state and the new nodes disappear. Each of the expanded nodes represent a similar news article to the expanded node. Figure 4.27 displays the result of a node expansion. Each node contains the keyword that best represents the news article on top of it. Each expanded node also shows the percentage of similarity between its news article and the one connected to it. When hovering between nodes, the news article displayed at the left (Figure 4.27) changes accordingly to the news article associated with the currently hovered node. All the features associated with the news article such the multimodal search and the image-based search are available to the user.

If the user wants to continue to expand the nodes in order to get similar news he can do so in the same way. It should be noted, however, that the similar news of the expanded node refers to the clicked news article node specifically. The reason the expanded news articles nodes refers to the clicked news article node specifically has to do with the fact that the user may be interested in a news article but hopes to consecutively continue on more relevant paths, and for that to happen, the expanded news article have to have some degree of dissimilarity. By providing more similar news articles of the same news article, it would offer the user more of the same, with little ability to provide different paths for the user to explore. In summary, when a node is clicked, the view expands five other similar news nodes specific to the clicked node and not the original node (green node), to offer similar exploration paths but with some degree of dissimilarity.

Figure 4.28 shows the result of expanding three different nodes. The nodes that are not close to each other are the nodes that were expanded. In the case of the figure, three different nodes were expanded. The red nodes that surround the blue node in the figure mean that the user has reached a point where they can not expand any further. All nodes that are not red can continue to be expanded.

4.6.2 Similar News Implementation And Workflow

This visualization, as opposed to other visualizations, is only triggered when a user clicks on the image. The implementation and worflow of this visualization will be detailed with the help of Figure 4.29.

When the user clicks on an image in the news grid, it sends a request to the JavaScript server passing the news ID appended with the news image order as an argument (step 1.). The JavaScript server then redirects the request to the Python server passing the exact same argument (step 2.). Since this visualization uses a tree like structure, the Python server uses a multimodal index to retrieve the five most similar news to the one passed as an argument. This process is repeated twice more for each retrieved news, meaning that the first 5 similar news will have a set of 5 similar news to that one and finally each of those news will have 5 more news set, setting the total number of news equal to 126(1 + 5*5*5). Since the data is hierarchical, a tree is created using the news ID as identifiers,

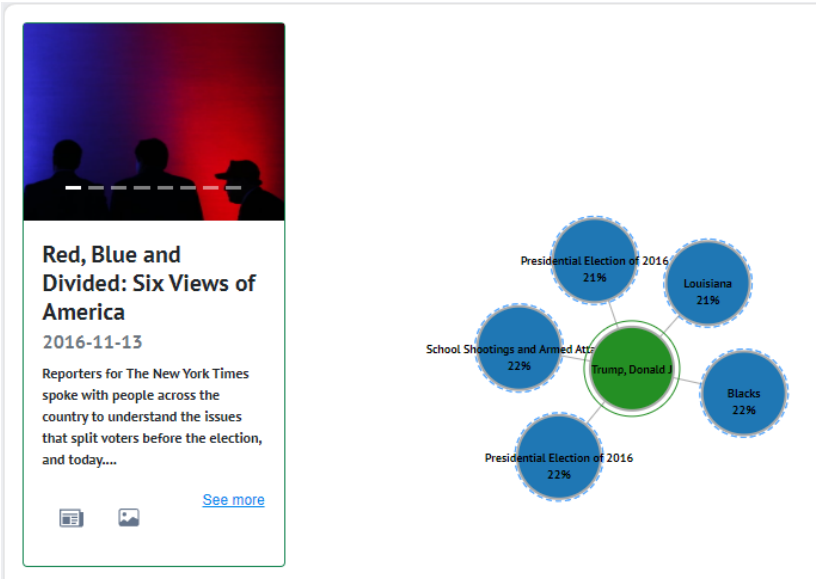


Figure 4.27: Similar News visualization after expanding the green node

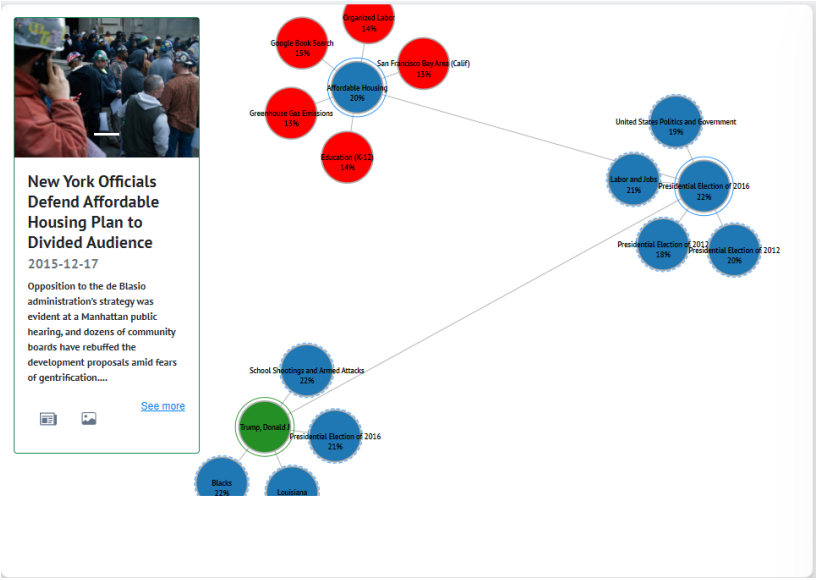


Figure 4.28: Similar News visualization after expanding three nodes

with the root being the news ID of the passed argument (Figure 4.30). This data is then sent back to the JavaScript server (step 3.). Since it is necessary to obtain the content of the news, it is also necessary to communicate with ElasticSearch in order to obtain the news content. In order to achieve this, the news IDs are extracted from the tree and stored in a list. Then, a request is made to ElasticSearch component using that same list (step 4.). Afterwards, the ElasticSearch sends the news content of each news corresponding to the news ID of the sent list (step 5.). Finally, the tree is reconstructed replacing the news IDs by their content instead of only their ids. This data is then sent to the user side and the visualization is loaded (step 6.).

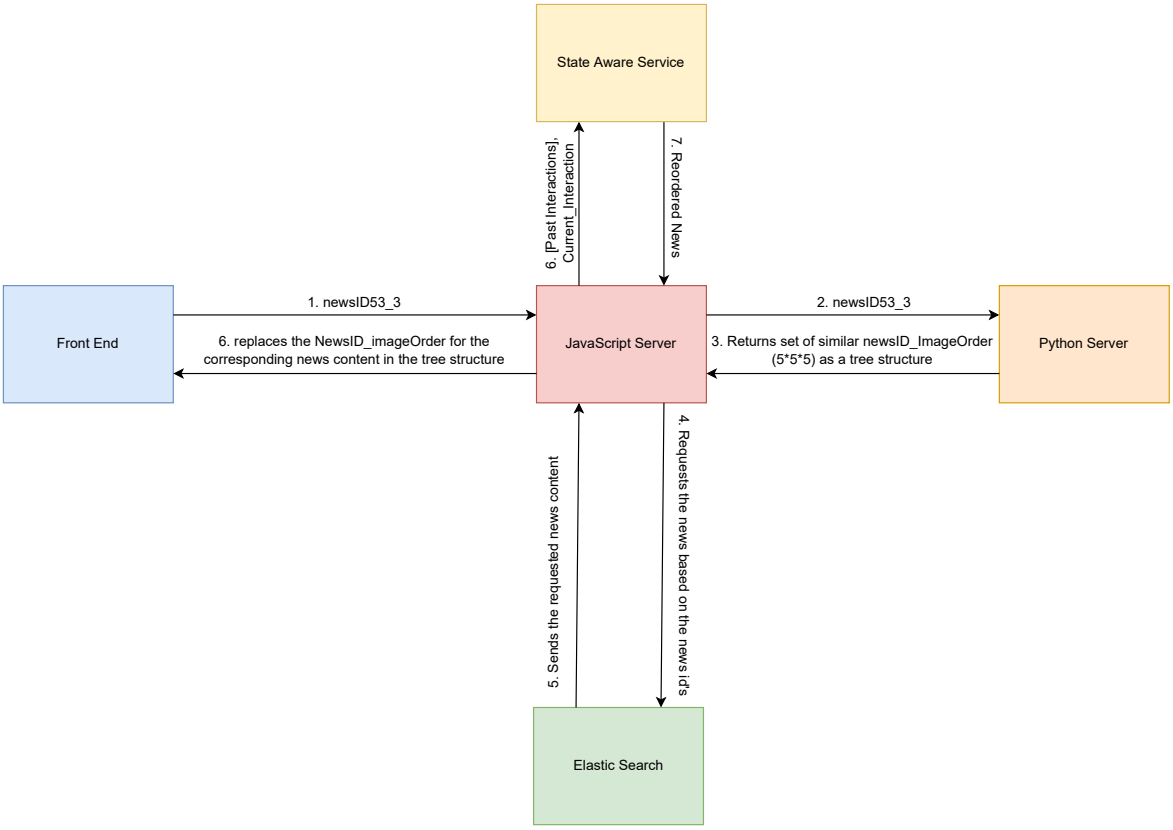


Figure 4.29: Similar News workflow

4.7 Bar Chart Visualization

Bar charts visualizations are one of the most common types of visualizations which mainly intend to represent simple x, y plots of data for numerical comparisons [34]. The bar chart visualization implemented and displayed in the user's interface makes no exception regarding the previous statement. Although bar charts do not offer any type of novelty to users, they are still intuitive and easy to understand [43], thus worth exploring with the right data.

The implemented Bar Chart visualization situates at the bottom of the time analysis

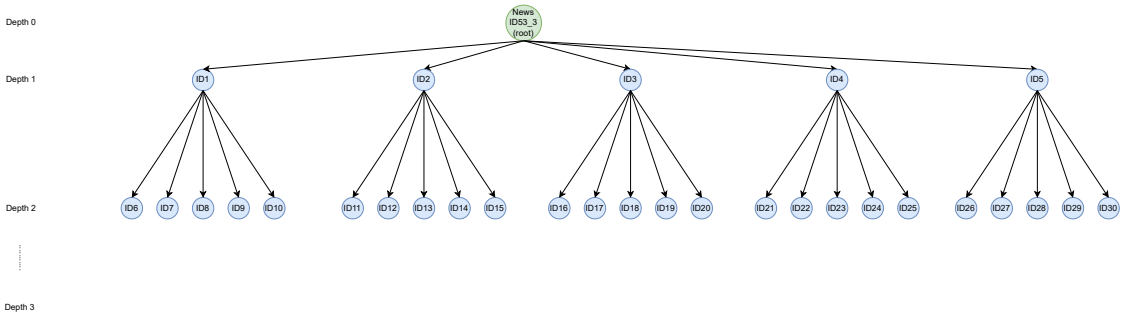


Figure 4.30: Tree like structure used in the visualization

tab (Figure 4.31). The user can choose to alternate between the bar chart visualization and the word graph visualization by clicking on the corresponding text that is located in the middle of the Figure 4.31.

4.7.1 Bar Chart User Interface

As previously mentioned, the user's interface makes use of bar charts visualizations. Figure 4.32 shows a close up of this exact visualization. In the Y axis of the figure situates the 10 most common keywords presented in the returned news set after searching for the word "Ukraine". The X axis of the figure represents the amount of news that contain a specific keyword. Regarding the interaction with the visualization, the user can click on any of the bars that belong to the visualization. If the user chooses to do so, the news feed refreshes, only showing the news that contain the keyword of the clicked bar. For example, if the user chooses to click on a bar corresponding to the "European Union" keyword in the Figure 4.32, then the news feed would only show news articles that contain "European Union" as a keyword. The result of that interaction is displayed in the Figure 4.33. As it can be seen, the time analysis visualization only shows 26 news spread across time. The Bar Chart visualization also updates by showing the 10 most common keywords of the 26 remaining news. If the user desires, he can continue to click on the bars of the bar chart to further refine and narrow his search and eventually find relevant news articles that please the user (i.e. in a turn-based fashion).

4.7.2 Bar Chart Implementation

The Bar Chart visualization is rather simple regarding the processing of data needed to load the visualization. After a request is made to the JavaScript server that requires the return of news content, for each news set to be returned, a list is created containing the all the keywords of the news ordered by frequency. This is possible because the news content has a field that specifies which keywords better represent a news article. After this process is completed, the 10 most frequent keywords in the entire news set are returned back to the JavaScript server. The Bar Chart visualization was designed with the D3.js library.

4.8 Word Graph Visualization

The goal of the Word Graph visualization is to use a graph like visualization to show the user the most frequent words of all the news shown when a user makes a search request and allow him to filter based on those words. It is often the case that by searching for a specific topic, different news articles have many of the same words specific to that topic, making those words frequent when considering a specific news set. If a specific word is frequent in a news set, it is possible that word is also relevant for the topic search in question. By allowing the user to filter the shown news set by clicking on the frequent words, the user is provided with the possibility of narrowing the amount of

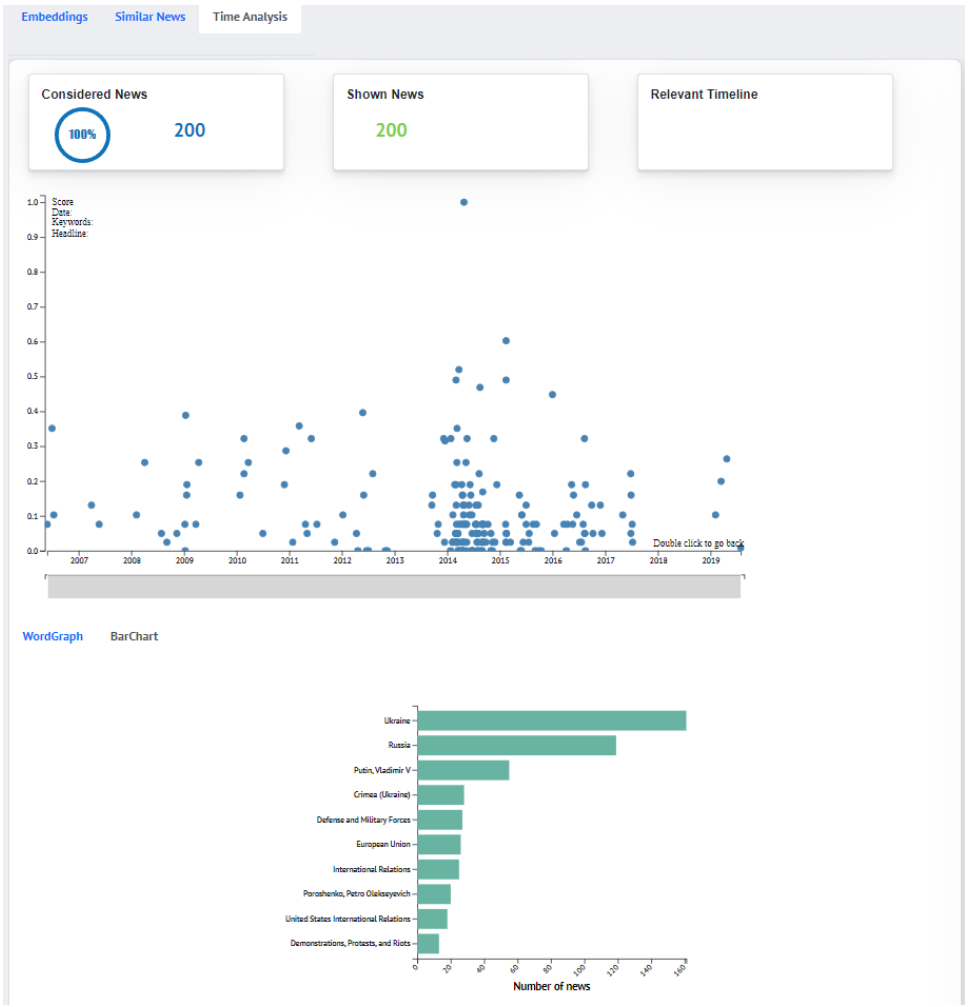


Figure 4.31: Time Analysis tab containing the Time Analysis visualization (top) and the Bar Chart visualization (bottom)

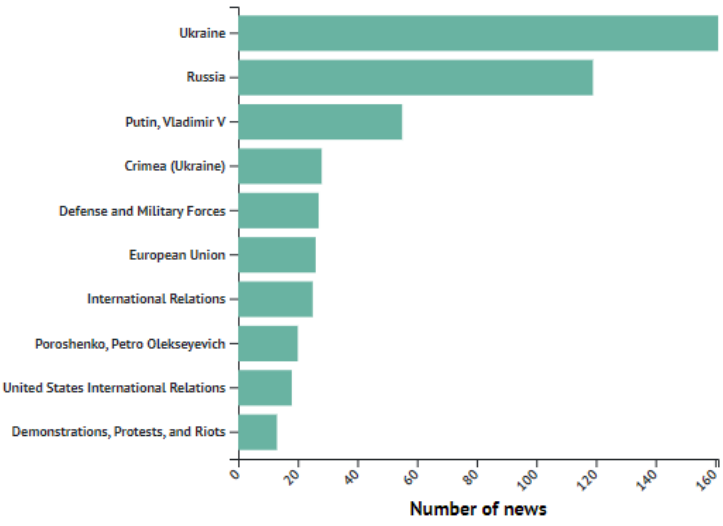


Figure 4.32: Bar Chart visualization after searching for the word “Ukraine”

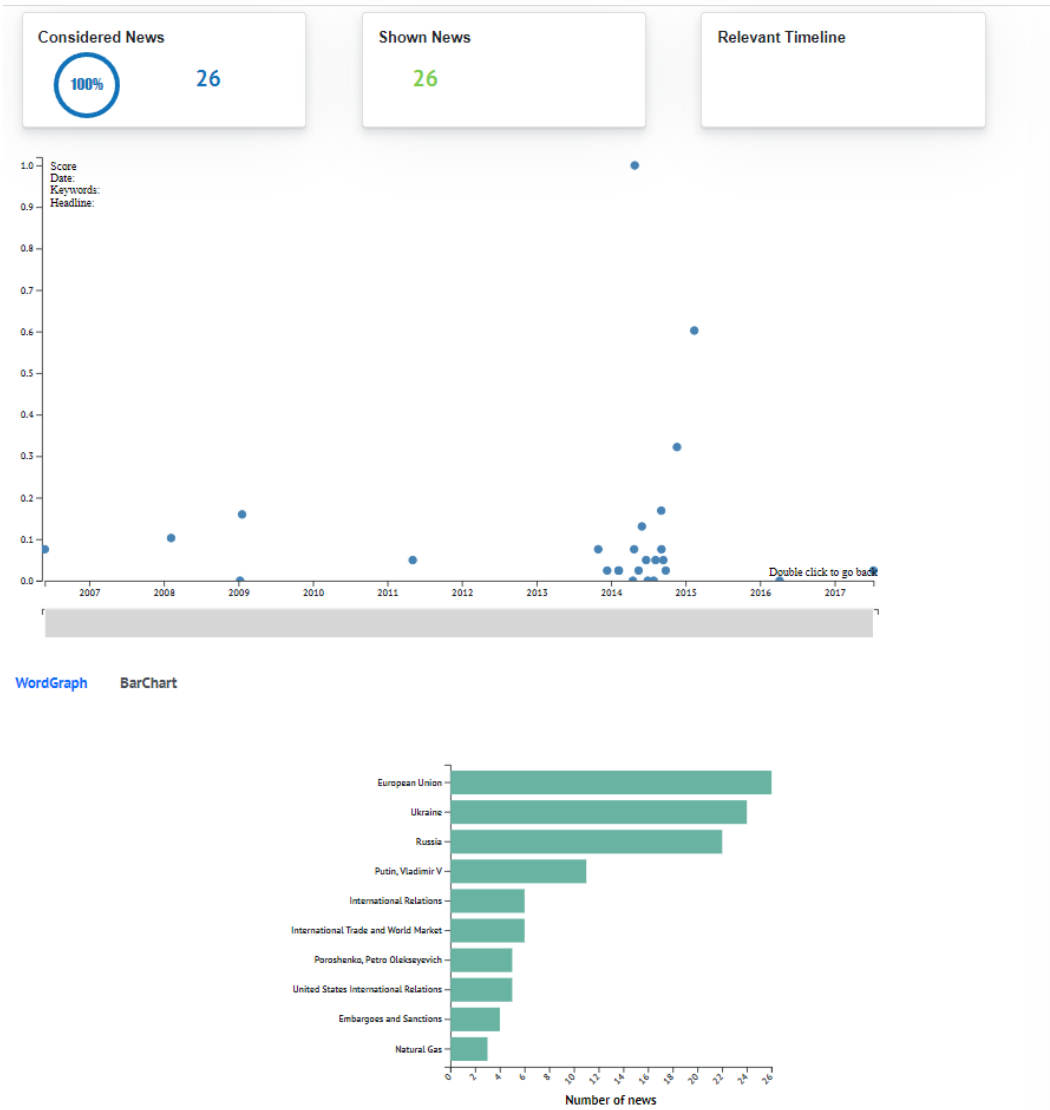


Figure 4.33: Image of the Time Analysis visualization (top) and the Bar Chart visualization (bottom) after clicking on the "European Union" Bar Chart keyword

news considered and hopefully provide the user with a more limited but relevant amount of news articles.

The Word Graph visualization is situated in the same place as the Bar Chart visualization, and can be accessed by clicking on the corresponding text located in the middle of the figure 4.31.

4.8.1 Word Graph User Interface

When the user makes a search query, such as a search on the search bar or clicks on one of the multimodal and image-based search buttons, a word graph is created. This word graph contains the top 50 most frequent words of all the news retrieved, which is 200. An example of this visualization after searching for the word "Ukraine" can be

seen in Figure 4.34. Words are represented by the blue nodes and the links represent a connection between nodes. One of the problems faced in the creation of this visualization was the elimination of non relevant keywords. Although there was an effort to reduce the amount of not so relevant words, there is still room for improvement. As it can be seen in Figure 4.34, the words "more", "Ms." and "other" are not much relevant words to filter by.

Regarding the interactivity with this visualization, when a user hovers the mouse on a node, the corresponding word of that node is highlighted in bold, as shown in the Figure 4.34. The word "minister" is highlighted. The user is allowed to zoom in or zoom out making the visualization closer or farther and is also allowed to drag and click the nodes presented in the visualization. If the user decides to click on one node, that node becomes red and the nodes (words) and edges that do not have a link to the clicked node become less visible, while the nodes and links that do have a connection between the clicked node stay the same. An example of this interaction can be seen in Figure 4.35. The word "Putin" is selected turning the node red, all the other nodes and links that not have a connection to the selected word become less visible.

When a node is clicked, it filters the news presented in the news feed, only showing news containing that specific word in their news article text. The Time Analysis visualization also changes based on the filtering of the news, only showing in the visualization the blue dots of the news that match the news feed, where the news are shown. Figure 4.36 shows the result of this filtering. As it can be seen, the time analysis visualization now only shows 110 news, which corresponds to the previous 200 news that have the word Putin in it. If the user decides that he no longer wants to filter by that specific word, he can click on the same node returning to the previous state (Figure 4.34), showing the initial 200 news.

After clicking on one node to filter the news by the specific word that the node represents, it is possible to further filter through the WordGraph visualization. The user can decide if he would like to click on one of the nodes that are still visible after the first click, further filtering the amount of news shown. For example, if the user decides to click on the node representing the word "Putin" to filter the news by that same word, reaching the same state as the one shown in Figures 4.35 and 4.36, and then the user decides to, for example, click on the word "Crimea", then the news would be filtered by both the word "Putin" and "Crimea". Meaning that the resulting news would have to have both words presented in their news text. The result of this interaction and filtering can be seen in Figures 4.37 and 4.38 respectively.

Lastly, if the user wants to undo this filtering, the user can click on the red nodes (the ones previously clicked) to go back to the previous state.

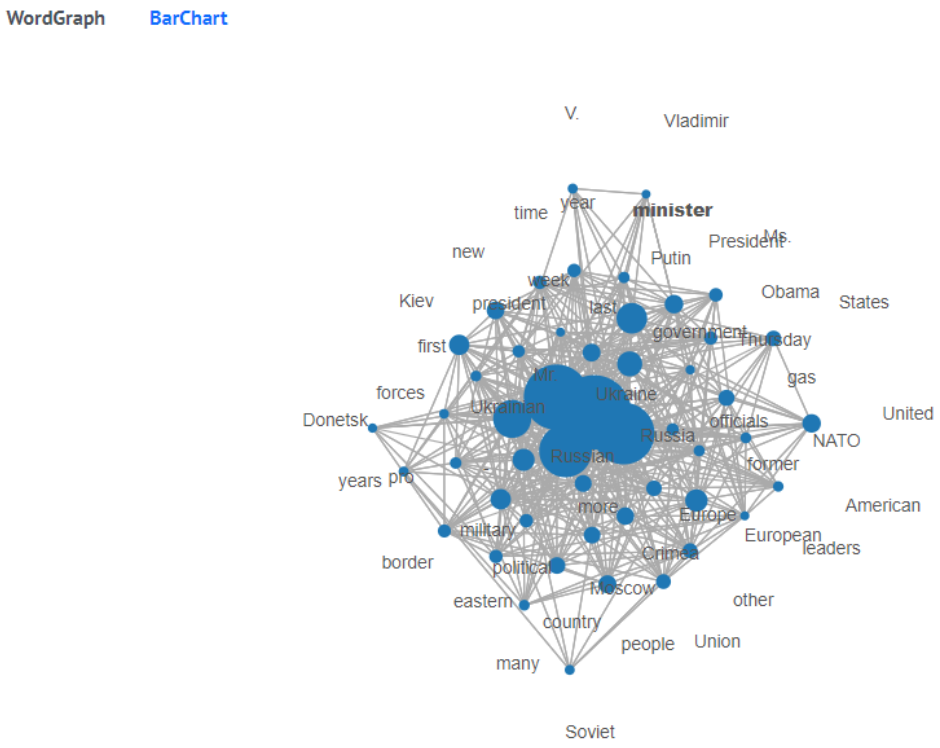


Figure 4.34: WordGraph visualization after searching for the word “Ukraine”

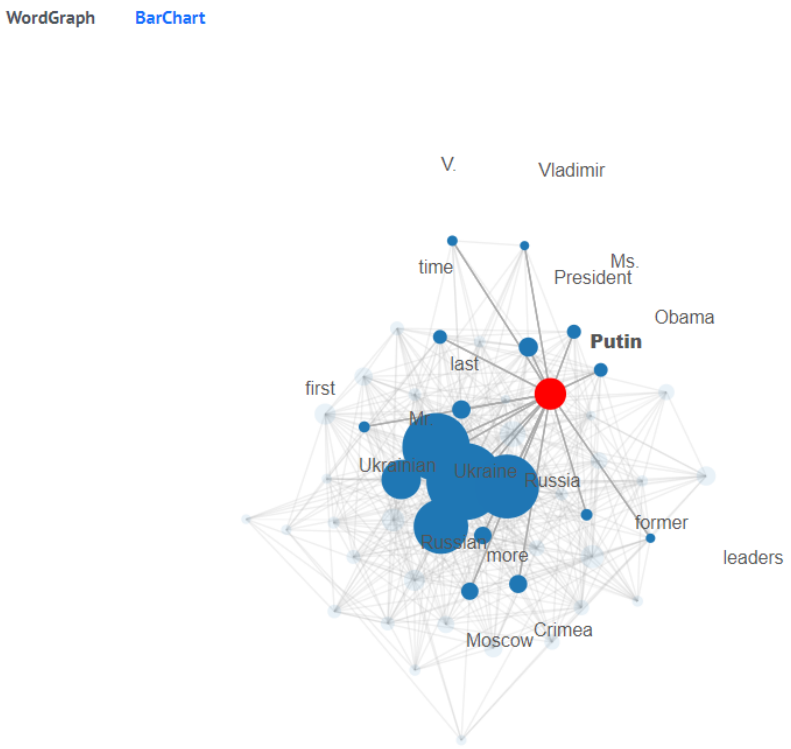


Figure 4.35: WordGraph visualization after click on the node with the word “Putin”

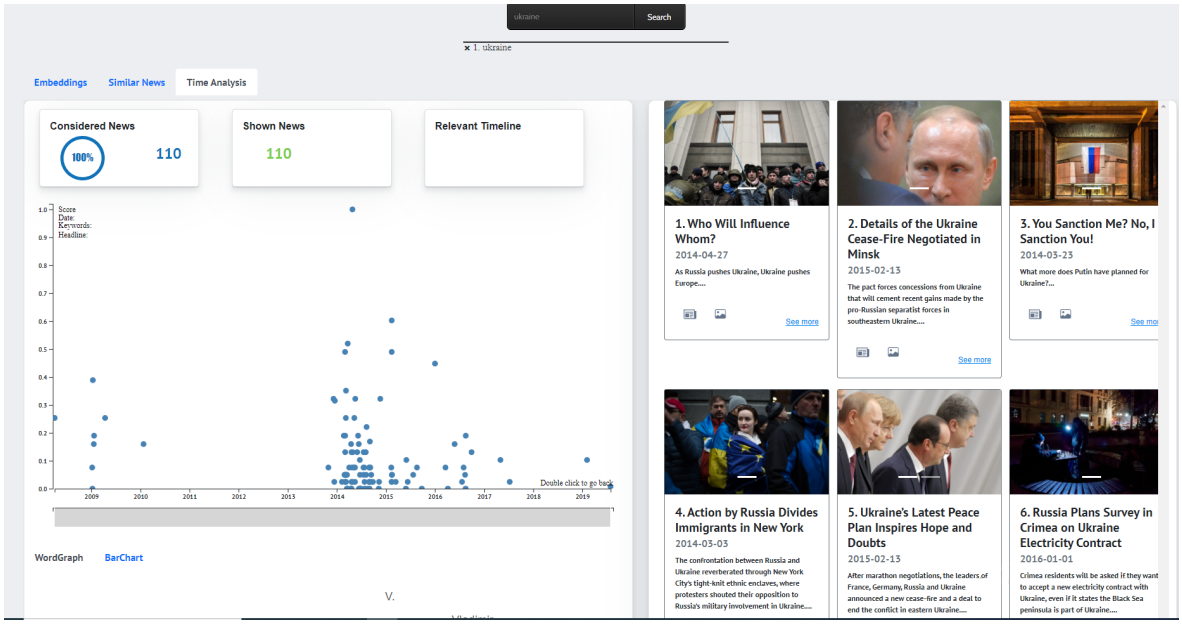


Figure 4.36: Result of the Time Analysis visualization (left) and the news feed (right) after filtering for the word “Putin” using the WordGraph visualization

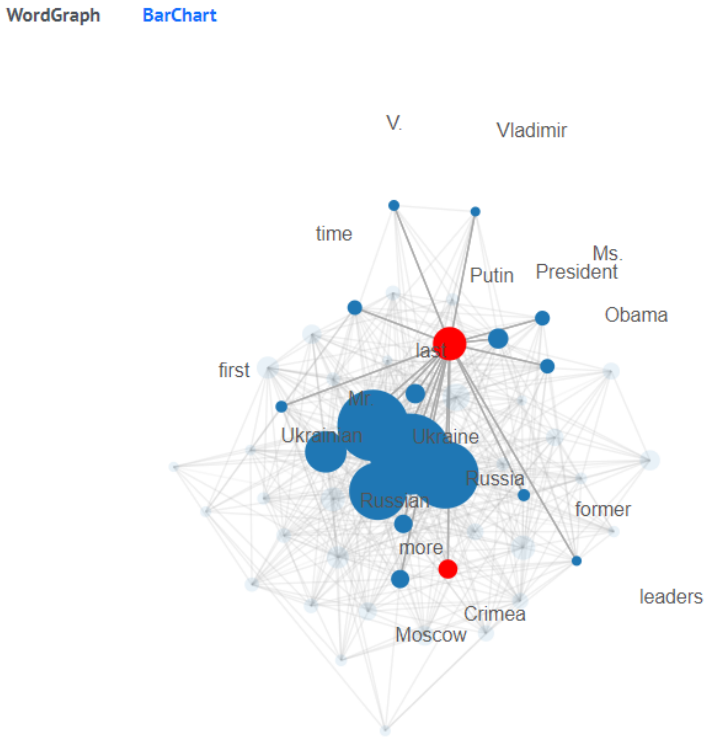


Figure 4.37: WordGraph visualization after click on the nodes with the word “Putin” and “Crimea”

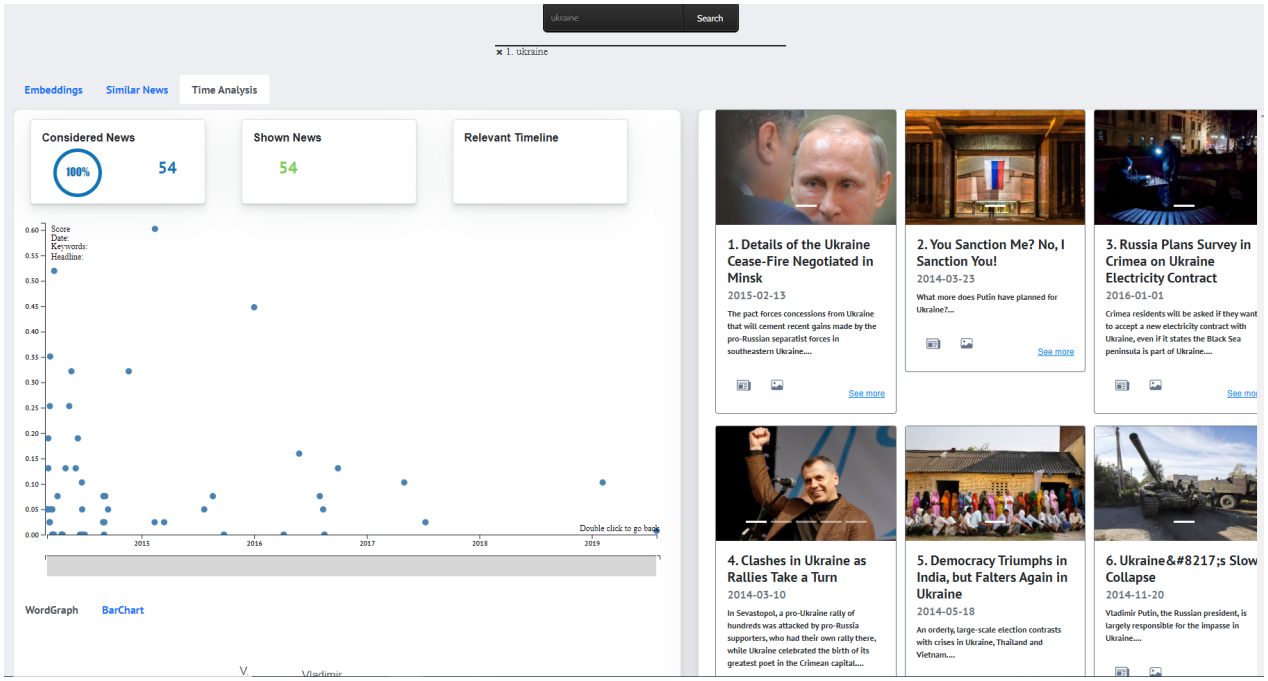


Figure 4.38: Result of the Time Analysis visualization (left) and the news feed (right) after filtering for the word "Putin" and "Crimea" using the WordGraph visualization

4.8.2 Word Graph Implementation

When a user makes a search request on the search bar, a request is made to the JavaScript server sending the searched word as a parameter. This request is then redirected to the Elasticsearch component that uses the searched word to retrieve the top 200 news articles and sends it back to the JavaScript server. Since this visualization uses graphs where nodes are represented by words and links represent a connection between two words, there is a need to have data structures that simplify the implementation of this visualization. It was, for this reason, decided to use two maps. A "WordToNewsMap" which stores as a key a word presented in a news article, and as a value, it stores the ID of the news that contain that specific word. For example, if out of 30 news, the word "Trump" is presented in the news with ID5, ID12, ID27, then the value of the word "Trump" would return the list -> [ID5, ID12, ID27]. An "EdgeMap" was also used, as the name indicates, this map will store the connections between words. How the "EdgeMap" stores connections will be explained further ahead.

It is important to mention that a lot of words in the news, although frequent, are not helpful for these types of visualizations that try to filter the news based on words. Words like "for", "on", "it"... are not helpful when trying to filter news by words as they don't offer anything relevant for the filtering process and consequently to the visualization. With this in mind, and due to the fact the news content has a field indicating the grammatical category of each word, it was decided to only keep words that are either adjectives, nouns or pronouns. This way, the risk of the visualization having words that are not relevant is

minimized.

Regarding the processing component of this visualisation, a naive approach was initially tested. This approach consisted in iterating all the words of all the news coming from the ElasticSearch that passed the constraints described previously, then the "EdgeMap" would store a link between all the words in the same news and repeat this process for every news left to iterate.

Finally, after the process is completed, only the 50 most frequent words are stored. The two maps (WordToNewsMap, EdgeMap) are then sent to the web application for the visualization to be loaded. If out of those 50 words, the "EdgeMap" has a connection between two words, a link is displayed in the visualization between those two words. This approach revealed to be sub-optimal because almost every word of the top 50 had a connection to all the other words of the top 50 which defeated the purpose of selecting words in the WordGraph. In order to counteract this problem, a different approach was taken. This approach was based on the InfraNodus [57] implementation which implemented a similar word graph that produced positive results. The approach still consisted in iterating all the words of all the news and filtering the words based on the same constraints, but the "EdgeMap" only stores edges between words that are up to three words appart. This approach allowed only words that were textually close to each other to have a link between them, also, the fact that it stores words that are up to three words appart considerably reduces the number of edges making it easier to understand and interact with the visualization. This approach produced much better results as the links between words made more sense.

4.9 System Session Storage

Since a user may make several search queries in a session in order to refine his search and due to the fact that the following searches take into account the previous searches, it is possible, for example, for a user to make a mistake in one of the searches, such as a spelling mistake and now wants to go back on his search. It is also possible that a user wants to go back to the state of previously made searches to follow a different refining path, such as trying other search queries while maintaining the state created by the previous searches. For example, a user can start by searching for the words "Putin" followed by "Ukraine" and "Invasion" in hopes to retrieve news about the 2014 Ukraine Crimea invasion. If the dataset used is recent, then it is likely that it would have both news from the 2014 Ukraine's Crimea invasion and the 2022 Ukraine invasion. By noticing this, the user would like to go back on the previous search and substitute the search word "Invasion" for "2014 Invasion" example, but without having to start from scratch and redo all the other queries. It was with this problem in mind that a feature was implemented to allow users to go back to a previous state created by past searches.

Users are allowed to go back to previous searches by clicking on the "x" next to the previous searched words. Figure 4.39 shows the interface search bar alongside the three

previous searched words. If a user decides to click on the "x" of the word "Election" presented in the figure, then that word would be removed from the words list and the news feed would restore all news articles and visualizations of the previous searched word, which is "White house" in this case. However, if the user clicked on the "White house" "x" the first time, then the system restores the state of the word before the clicked one, which would be "Trump".

When a user makes a request, a large amount of data and resources have to be processed by the different servers and since when a user decides to go back to a previous state, the data that feeds the visualizations and the news feed was already sent before, the data sent on each search request is stored during a user's session in order to save server resources.

Due to the fact that for each search request made the data sent to the web application is stored, the time that it takes to refresh the news feed and the corresponding visualizations with the corresponding data is negligible. As the web application does not have to make requests to other components to process any kind of data.

In conclusion, this feature was implemented to allow users to go back on their searches if they notice that certain searches did not provide the expected results, or the user simply wants to continue searching for some topic but refine it in other way.

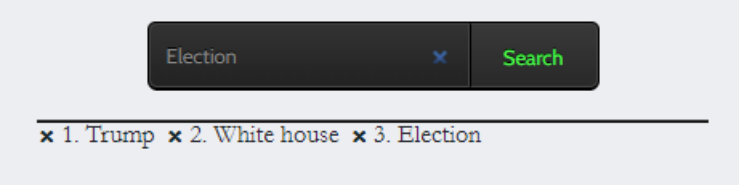


Figure 4.39: Search bar and previously searched words

5

Evaluation

The evaluation of the system's usability is a fundamental part of the entire process of development of software system's, specifically systems that provide interfaces that can be used by a large quantity of users. Through the systems evaluation it is possible to obtain feedback concerning certain aspects such as the users satisfaction with the system, the learning curve (i.e., whether it is necessary for the user to obtain a high level of experience until mastering the system) and also to understand if the system meets the expected functions defined at the time of development. In order to evaluate the system's usability, typically test sessions are carried out with users, these users usually follow a set of predetermined tasks where their results are later evaluated in order to obtain a conclusive result regarding the evaluation of the system. In general, after the tests, one or more questionnaires are filled out by users. This method of evaluating is seen as the most reliable and valid [21].

The usability component of the system, was evaluated by asking users to test the system's interface. After testing each of the main components of the interface the user would fill part of a questionnaire which only had questions related to the tested component. As for the tasks performed during the test sessions, they were divided according to the interface logic. Since the interface contains 3 tabs and each tab contains one or more visualizations where different visualizations within the same tab are linked, it was decided to test each tab individually. After testing each of the different tabs, the user was asked to fill out a questionnaire of the respective tested tab. At the end of the questionnaire the user is asked to fill out The System Usability Scale (SUS). As the name implies, this type of questionnaire serves to test the usability of the system in use. This questionnaire was created in 1986 by John Brooke [11] and consists of a 10 item questionnaire with five response options for respondents; from Strongly agree to Strongly disagree, depending on the weighting of the respondents' agreement to the 10 statements about their perception of the system's usability. Due to the fact that all the questions and tasks were written in Portuguese, table 5.1 shows the original questions written in English and the translation shown to the users in Portuguese.

Nº	Original	Translation
1.	I think that I would like to use this feature frequently.	Penso que gostaria de utilizar este sistema com frequência.
2.	I found the feature unnecessarily complex.	Achei o sistema desnecessariamente complexo.
3.	I thought the feature was easy to use.	Pensei que o sistema era fácil de utilizar.
4.	I think that I would need the support of a technical person to be able to use this feature.	Penso que precisaria do apoio de uma pessoa técnica para poder utilizar este sistema.
5.	I found the various functions in this feature were well integrated.	Descobri que as várias funções deste sistema estavam bem integradas.
6.	I thought there was too much inconsistency in this feature.	Achei que havia demasiada incoerência neste sistema.
7.	I would imagine that most people would learn to use this feature very quickly.	Imagino que a maioria das pessoas aprenderia a utilizar este sistema muito rapidamente.
8.	I found the feature very cumbersome to use.	Achei o sistema muito complicado de usar.
9.	I felt very confident using the feature.	Senti-me confiante ao utilizar o sistema.
10.	I needed to learn a lot of things before I could get going with this feature.	Precisava de aprender muitas coisas antes de poder avançar com este sistema.

Table 5.1: SUS questions originally written in English and translated to Portuguese

The SUS questionnaire is seen as one of the most popular with the purpose of evaluating the usability of software systems [47].

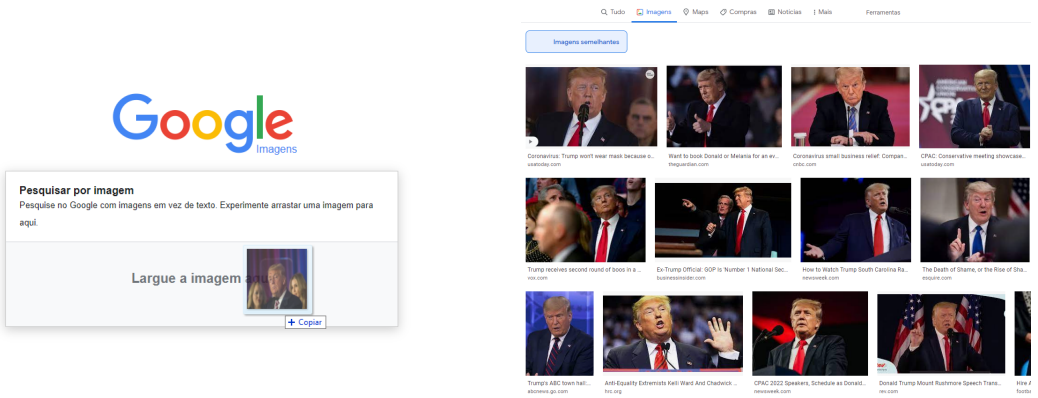
5.1 User Characterization

The total user sample consists of 13 volunteers, all of whom tested the interface and filled out the questionnaires as intended. Since the system is available on an online platform through a link, it was possible to perform tests remotely as well as in person. About 8 volunteers performed the tests in person, while the remaining 5 people performed the tests online. 10 of the users are between the age group of 20 to 29 and 3 of the users are between the age of 45 to 60. 10 of the users are male while the remaining 3 users are female.

One of the concerns regarding the tests was to understand the level of familiarity with tools that offer users functionalities similar to those offered by the system under test, since this can influence the test results. Tools such as google images, allow users to obtain similar images to one of their choice. For example, if a user wants to obtain similar

images in relation to another image, google images provides this capability. Figure 5.1 shows an example of the previous described interaction. Subfigure 5.1(a) shows a possible search based on a Trump image using google images. Subfigure 5.1(b) shows the resulting search. Since the system under test offers users the "search by image" feature, described in section 4.3.2, allowing users to retrieve news articles based on images, the interaction is quite similar. With respects to the familiarity of the users with tools such as google images, 100% of the users admitted to use or having used google images before.

Users were also inquired about their experience with the use of computers which 100% of the participants admited to use computers regularly. Lastly, users were asked whether they usually use news sites which 76% admited to use regularly.



(a) Example of a search based on an image using google images

(b) Resulting similar images based on the image search of figure 5.1(a)

Figure 5.1: Example of a Google Image search using an image (fig 5.1(a)) and its results (fig 5.1(b))

5.2 User Study

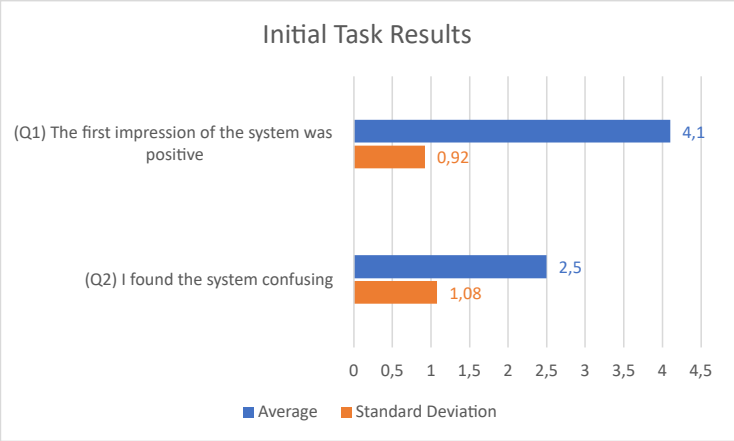
As previously mentioned, user tests were carried out with the purpose of evaluating the system. These tests are compromised of several tasks which were divided based on the main components of the interface. The goal of the tasks was to evaluate the news feed and each visualization tab separately. This means that each tab (Time Analysis, Similar News and the Embeddings tab) had a set of tasks with the purpose of testing the visualization(s) contained therein. Table 5.2 enumerates the different tasks that will be detailed further ahead in the document, it also shows the number of tasks of each task group, the number of questions that users had to respond to and at last, a brief description explaining the procedure of each task group.

By testing each interface component separately it will then be possible to draw conclusions about each of the components and understand their strong and weak points. The user tests consisted in questions to which users would answer through the use of likert scales and sets of tasks to evaluate if users are able to complete them. A likert scale is a

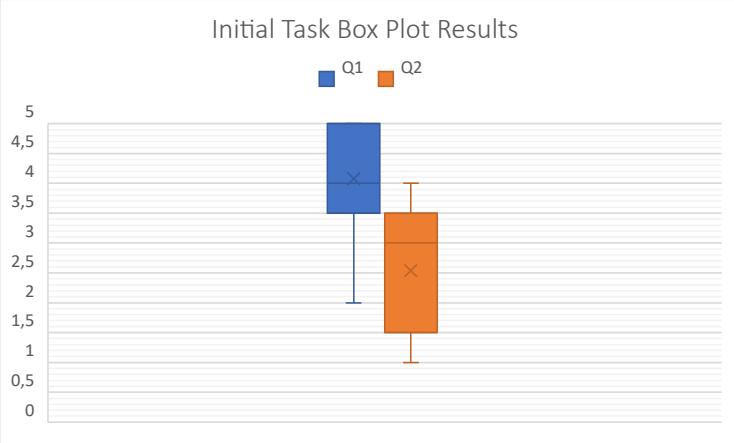
rating system used in questionnaires that is designed to measure users attitudes, opinions, or perceptions in regards to a specific question. The likert scales used in the user tests were coded numerically, in which case the numerical values represents an opinion or perception, such as 1 = strongly disagree, 5 = strongly agree [49].

Task	Nº Tasks	Nº Questions	Description
Q - Initial Task	2	0	The initial task compromised on allowing users to freely use the system as they please so users can gain some degree of familiarity with the system.
A - Basic Tasks	2	4	After the user becomes slightly acquaintanced with the system, the user is asked some very basic tasks and questions with the purpose of understanding if users are able to understand the different components of the interface.
B - Intermediate Tasks	3	2	Users were asked to execute tasks that do not involve the visualizations, such as performing a system reset and refine the search to obtain similar news articles through the use of the multimodal and image based search buttons.
C - Advanced: Time Analysis	5	4	Users completed tasks to understand if the design and usability of the Time Analysis tab was satisfactory, followed by questions to conclude if users were able to correctly interpret what the components of the visualizations represent.
E - Advanced: Similar News	4	4	The user is first asked to perform tasks to understand the user's ability to interact with the Similar News visualization, afterwards, the user answers a set of questions in order to obtain more information.
F - Advanced: Embeddings	5	5	The user completes a set of tasks. Some questions are asked to obtain more information about the visualization, its intuitiveness, the amount of information shown and the interacted components.

Table 5.2: Enumeration of the different tasks performed in the user tests with the respective number of tasks, questions and brief description of what the tasks include



(a) Initial tasks likert scale results



(b) Initial task box plot results

Figure 5.2: Initial task results (Q1 and Q2)

5.3 Q - Initial Tasks

The initial task compromised on allowing users to freely use the system as they please so users can gain some degree of familiarity with the system. There is not a specific task to be made, during this task the user is supposed to explore the visualizations and the news feed so the user can get acquaintance with the system. Afterwards, the user is asked to answer two questions (Q1 and Q2) through the use of a likert scale, namely: "*The first impression of the system was positive*" and "*I found the system confusing*". The results of these questions can be seen in Figure 5.2. In general, users showed a positive initial impression of the system as it can be seen in Subfigure 5.2(a). The first question (Q1) obtained a mean score of 4.1/5 with a standard deviation of 0.92. Subfigure 5.2(b) displays the box plot of the variation of observed data by means of quartiles. The depicted box plot also shows that the Q1 question obtained a median of 4 while the interquartile ranges from 3.5 to 5 with a lower whiskers ranging from 3.5 to 2.

With respects to the second question (Q2), users found the initial impression of the

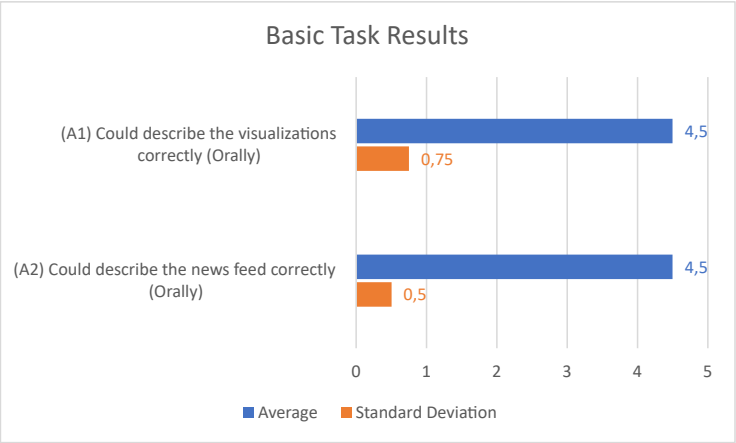
system to be slightly confusing. Subfigure 5.2(a) shows that the second question scored a mean of 2.5/5 with a standard deviation of 1.08. The box plot presented in subfigure 5.2(b) also supports the premise that users found the system confusing at first. The interquartile group depicted in the figure ranges from 1.5 to 3.5, while the lower and upper whiskers ranges from 1 to 1.5 and 3.5 to 4 respectively. Also, as it can be seen in the figure, approximately 50% (3rd and 4th quartile groups) of the participants scores is between 3 and 4. Additionally, 23% of the users revealed that the reason they found the system to be slightly confusing was probably due to the fact that they are not used to work with systems that use these types of visualizations and that there is a degree of unfamiliarity with these types of systems. Furthermore, 30% of users revealed that it took them a while to understand what was the purpose of some visualizations, this sentiment was specifically prevailing regarding the Embeddings visualization. The results of the second question accompanied with user feedback indicate that in general, users need some type of experience with the system before understanding its purpose, more specifically, the visualizations purpose. This lack of initial understanding translated in the user finding the system more confusing.

5.4 A - Basic Tasks

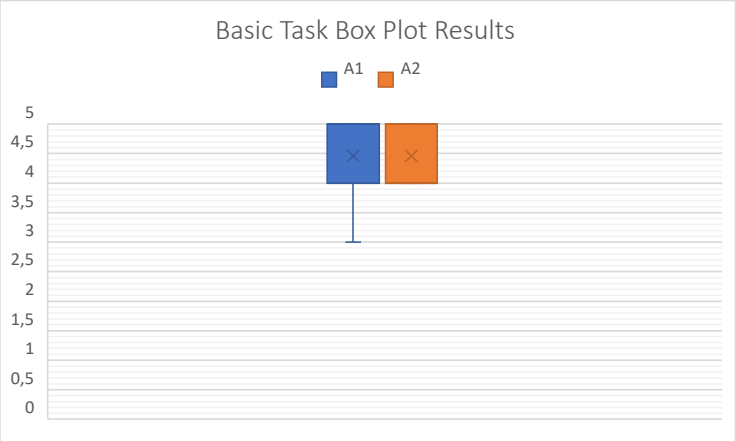
After the user completes the first initial task and becomes slightly acquaintanced with the system, the user is asked some very basic tasks and questions with the purpose of understanding if users are able to understand the different components of the interface (i.e., if users are able to understand which part of the system corresponds to the visualizations and which parts correspond to the news feed) and their opinion about the overall layout of the system. The questions and its results about this task specifically can be seen in Figures 5.3, 5.4 and 5.5. Figures 5.3 and 5.5 show the results of questions answered through the use of likert scales (questions A1 to A2 and A5 to A6). Figure 5.4 shows the percentage of users that were able to complete a certain task without external help.

The two first questions of this group (A1 and A2) are meant to understand if users are able to comprehend how the system is divided. Overall, users had a very positive understanding of what the main parts of the system represent, as the mean score of both questions (A1 and A2) presented in Subfigure 5.3(a) are of 4.5/5 with a standard deviation of 0.75 and 0.5 respectively. These questions were asked to the users orally but then scored between 1-5 depending on the answer, with a completely wrong answer receiving a score of 1 and a perfect answer receiving a score of 5. Subfigure 5.3(b) shows that the A1 question obtained a median of 5 and an interquartile ranging from 4 to 5 with a lower whiskers ranging from 3 to 4. With respects to the A2 question, 100% of the participants scores is between 4 and 5 .

The next two tasks (A3 and A4), described in Figure 5.4 have the intent of understanding whether the users are able to perform the simplest tasks of the system, such as searching for a specific topic using a search bar, find a certain result of that topic, scroll



(a) Basic tasks likert scale results



(b) Basic tasks box plot results

Figure 5.3: Basic task results (A1 and A2)

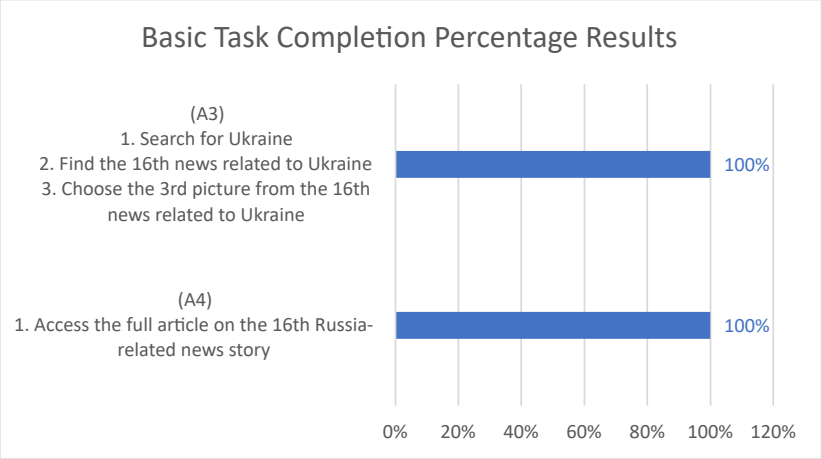


Figure 5.4: Completion percentage of A3 and A4 tasks

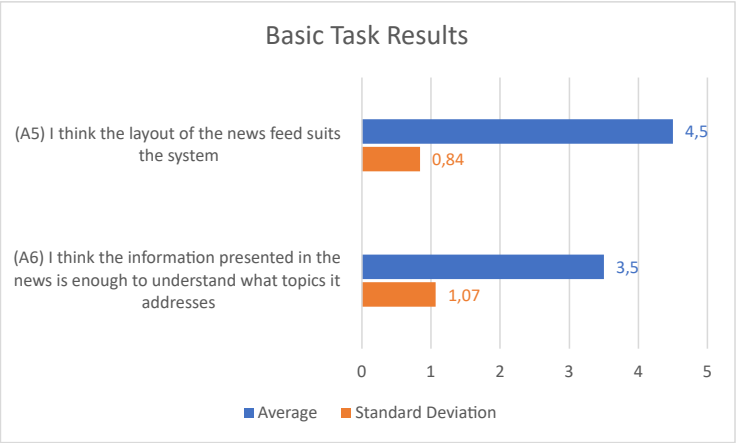
through the news articles images, etc. The web application interface follows the typical layout of web sites that allows for searching and viewing of the results and for that reason, it is very likely that users already have a high degree of familiarity with tasks of this kind and a poor result of these tasks would possibly point to a serious flaw in the design of the system. Regarding the results of the A3 and A4 tasks, 100% of the participants were able to complete the tasks without external help. In general, users showed ease in completing these tasks, stating that it follows the typical search manner usually found on web sites such as the Google [1] search engine.

Lastly, the two last questions of this group, A5 and A6 represented in Figure 5.5 make reference to the designed layout of the system and reference to the users' opinion about the information shown in the news feed. Regarding the A5 question, Subfigure 5.5(a) shows that users found the layout to be appropriate for the system in question as the results average a mean of 4.5/5 with a standard deviation of 0.86. Furthermore, Subfigure 5.5(b) exhibits that only one of the participants gave a score of 2 while the rest of the participants scores are between 4 and 5. As for the A6 question however, users noted that in general, the title and snippet of a news article presented in the news feed is often not enough to understand which topics the news article address. Some users reported that the title and snippet of a news article "*do not give an idea about the specifics of a news article.*" and that they would need more information in order to solve this problem. This difficulty is mirrored in the results, since the results of this question (A6) presented in Subfigure 5.5(a) obtained a mean score of 3.5/5 with a standard deviation of 1.07. Also, as it can be seen in Subfigure 5.5(b), the A6 question obtained a median of 3 with an interquartile group ranging from 2.5 to 5 and a lower whisker ranging between 2 and 2.5. Approximately 50% of users which results from the sum of the 1st and 2nd quartile gave a score between 2 and 3.

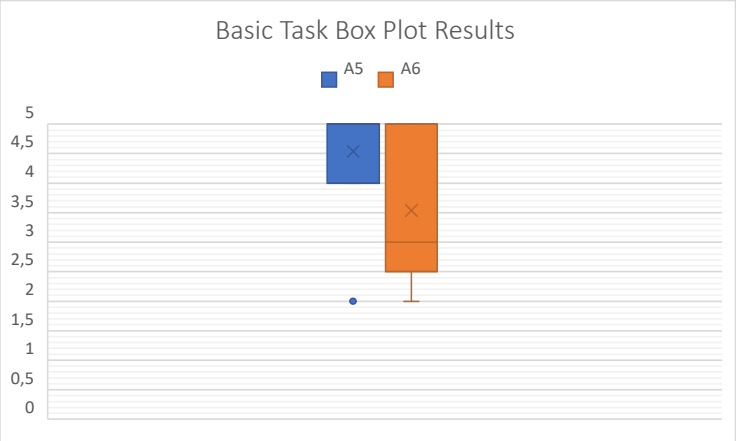
5.5 B - Intermediate Tasks

Users were also asked to execute other actions which are also still related to basic system tasks that do not involve the visualizations, the reason for this has to do with the fact that the users first need to be able to dominate basic tasks before interacting with the visualizations. The first three tasks (B1, B2 and B3) are meant to evaluate if the users are able to complete them without any kind of help. Users were asked to: (B1) perform a full or partial system reset, (B2) refine the search to obtain similar news items in relation to the fifth news item, and to (B3) refine the search to obtain similar news items in relation to the fifth news item but through images. The results to these questions can be seen in Figure 5.6. Figure 5.6 shows that 85% of the users were able to complete the task B1 without help, 92% of the users completed the B2 task without help, and lastly, 92% of the users were able to complete task B3 without any help. The results of these tasks can

[1] https://www.google.pt/?hl=en-EN



(a) Basic tasks likert scale results



(b) Basic tasks box plot results

Figure 5.5: Basic task results (A5 and A6)

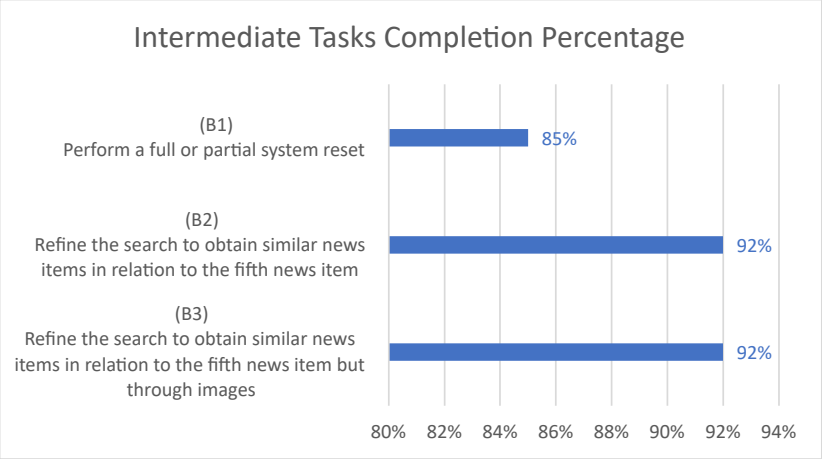


Figure 5.6: Completion percentage of B1, B2 and B3 tasks

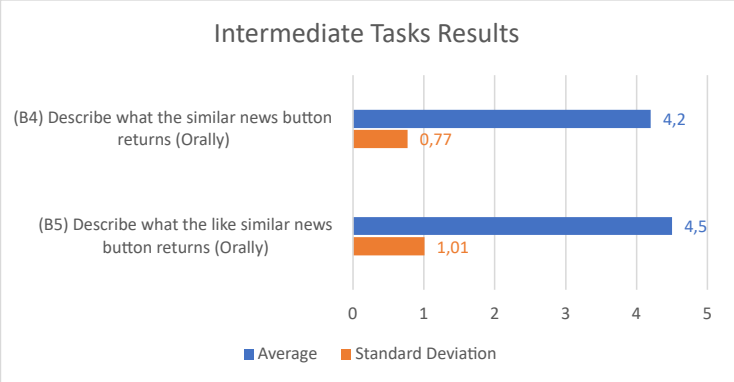
be seen with a positive connotation as most of the users were able to complete the tasks with little experience with the system.

The two next questions (B4 and B5) of this group were made in order to understand if the user was able to correctly describe what each button (used in tasks B2 and B3) return, as it is possible that although the user was able to complete the tasks, they might not have understood exactly what each button returned. The results of these questions can also be seen in Figure 5.7. Subfigure 5.7(a) shows that the mean scores of the questions B4 and B5 are of 4.2/5 and 4.5/5 respectively in regards to the mean, with a standard deviation of 0.77 and 1.01. Subfigure 5.7(b) displays the box plot of the two questions. As it can be seen, with respects to the B4 question it obtained a median of 4 with an interquartile between 4 and 5 with a lower whiskers ranging from 3 to 3.5. Regarding the B5 question, it obtained even better results, although the median is the same at 4 its interquartile ranges from 4 to 5 with a lower whiskers ranging from 3 to 4 with an outlier score of 2.

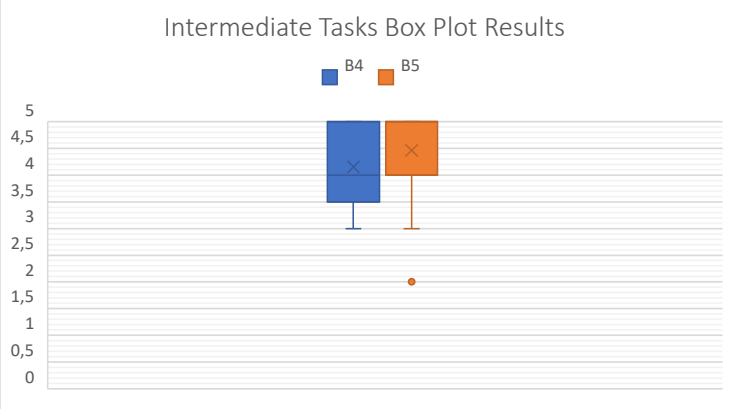
The results show that in general users were able to correctly describe what the multimodal and image-based search button return. The most common "mistake" users would make when describing their understanding of what the button returns is that users would assume that when returning the similar news articles, it would only take the text of the news article into account, when in fact, it takes into account both the text of the news article and the image present in the article. Regarding the image-based search button, users in general correctly described what the button returns, users generally would give a mostly correct answer but approximately 40% of users displayed some sort of unsurety in their answer, something that was not noticed with the multimodal search button although it scored a lower result. Users also reported that the icon used for the image-based search button was not exactly the best one and a more fitting one might have helped their understanding.

5.6 C - Advanced Tasks: Time Analysis Tab Testing

Since the interface contains several components that need to be tested, it is also necessary to test the visualizations. As mentioned earlier the test tasks were separated according to the logic of the system and after the completion of one of the tasks the user answers a questionnaire regarding the tested part. Since the visualizations are separated logically according to the tabs, each tab was submitted to tests. The first tested tab was the Time Analysis tab. Users were asked to complete certain tasks to understand if the design and usability of the Time Analysis tab was satisfactory. The part of the questionnaire related to the Time Analysis tab is separated into practical tasks, where the goal is to conclude if users are able to handle the different visualizations in this tab as it is supposed to be, followed by questions in order to understand if users were able to correctly understand what the components of the visualizations represent. The first five tasks and its results are represented in Figure 5.8. As shown in the figure, most of the users were able to complete



(a) Intermediate tasks likert scale results



(b) Intermediate tasks box plot results

Figure 5.7: Basic task results (B4 and B5)

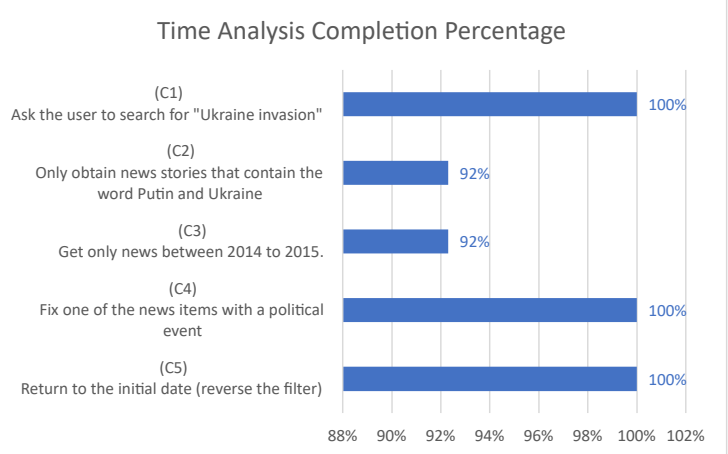


Figure 5.8: Completion percentage of C1 to C5 tasks

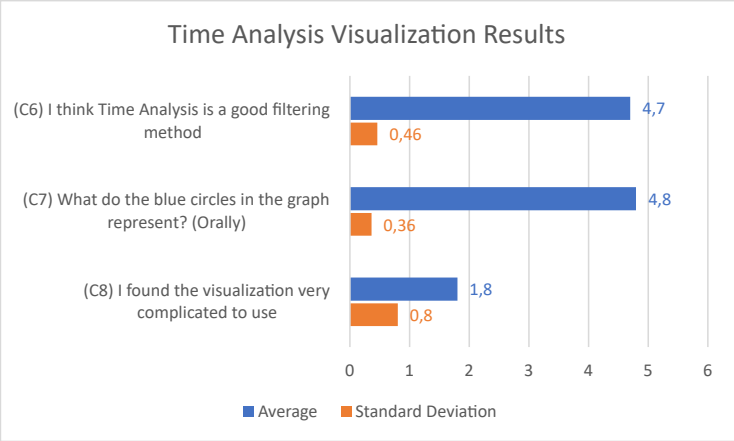
the tasks. The minimum completion percentage score of the tasks is of 92%, while 3 out of the 5 tasks had a completion percentage of 100%. Although these tasks involve users having to interact with visualizations that are linked to each other, meaning that the interaction with one visualization affects other visualizations, which could in theory affect the degree of difficulty of the tasks, the results show that overall the percentage of users that were able to complete the tasks is high.

After the completion of the previously described tasks, users were questioned about their understanding of the visualizations and their components to access if their understanding is in accordance with what is represented in the visualizations. Users were also questioned about the utility of the visualizations. The questions and their results can be seen in Figure 5.9. Figure 5.9(a) shows that the mean score of the C6 question is of 4.7/5 with a standard deviation of 0.46. Figure 5.9(b) also shows that all the participants scores ranges from 4 to 5 with a median of 5. With these results it is possible to conclude that users perceive the visualizations in the Time Analysis tab as a positive method of filtering information. With respects to the C7 question, with an average score of 4.8/5 and a standard deviation of 0.36 (Figure 5.9(a)) it is also possible to conclude that users were able to identify that the blue dots in the Time Analysis visualization represents news articles. This conclusion is also supported by the box plot presented in Figure 5.9(b) which shows that the amount of users who were not able to perfectly describe what the blue circles represent and thus obtain a score of 5 are an outlier. Regarding the results of the C8 question, it obtained a mean score of 1.8 and a standard deviation of 0.8 (Figure 5.9(a)). Figure 5.9(b) shows that users scores ranges from 1 to 2, with an outlier score at 4. The median of this question is of 2. The results point in the direction that overall the Time Analysis tab is not perceived as being difficult to use. Lastly, with regards to the C9 question, only 50% of the participants could correctly answer what the words shown in the WordGraph represent. The options and the correct answer for this question are shown in Figure 5.10.

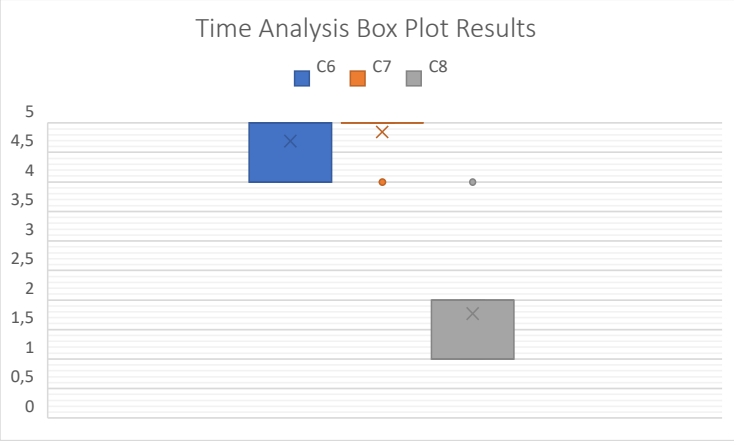
The results of this questions show that there is some difficulty regarding the comprehension of the WordGraph. Although 83% of the users were able to complete the C2 task which made use of the WordGraph, only 50% of the participants correctly answered the C9 question, which shows a contrast between usability and understanding and thus the need to evaluate both aspects.

5.7 E - Advanced Tasks: Similar News Tab Testing

Another of the tabs present in the interface that the user interacts with is the Similar News tab. The tasks performed in order to test the visualization present in this tab were processed in the same way as the Time Analysis, that is, the user is first asked to perform 4 tasks related to this tab in order to understand the user's ability to interact with the visualization in question, afterwards, the user answers a set of questions in order to obtain



(a) Time Analysis tasks likert scale results



(b) Time Analysis tasks box plot results

Figure 5.9: Time Analysis task results (C6 to C8)

- **Question C9:** What do the words in the WordGraph represent?
  - Keywords present in the news articles.
  - Random words present in the news articles.
  - News articles topics.
  - Frequent words in the news articles.
  - Other.

Figure 5.10: Question C9 with corresponding options and correct answer (filled dot)

more information about, amongst other things, the intuitiveness and perception of the user during the execution of the tasks.

Because the Similar News visualization is a rather simple one, as users do not have much to interact with other than expanding, collapsing and dragging nodes or zoom in and out on the visualization screen, the tasks are relatively short and simple, meaning that it is not expected for a big portion of users not to be able to correctly execute the tasks. The tasks and its results can be seen in Figure 5.11. As depicted in the figure, the results are in line with what is expected from the users. 100% of the users were able to complete all the first two tasks without any external help. Only 8% of the users were not able to complete task E3, which is similar to the E2 task but the user has to expand and search more nodes to find the news article with the keyword "House of Representatives". Nevertheless the completion percentage of the E3 task is still very high. Finally, 85% of the users were able to complete task E4, which requires users to collapse only the initial node, which is the only node with the color green.

With regards to the questionnaire filled out by users regarding the Similar News visualization, its results are presented in Figure 5.12. With regards to E5 question, which asks the users to rate the intuitiveness of the system, Subfigure 5.12(a) exhibits that the mean score was of 3.7/5 with a standard deviation of 0.91. Also, Subfigure 5.14(b) exhibits that all the participants scores ranges from 3 to 5, with 3 being the median. In general, users showed ease and reported that although the interactivity with the visualization was straightforward and not complex, 34% of users also reported that they had trouble understanding what was the purpose of the visualization, which translated in a lower score when rating its intuitiveness. Question E6 which asks users if they think that the keywords are enough to describe the words was the question that obtained the lowest results of this task, it averaged a score of 2.6 with a standard deviation of 0.74, its interquartile ranges from 2 to 3, with a upper whiskers ranging from 3 to 4 and a median

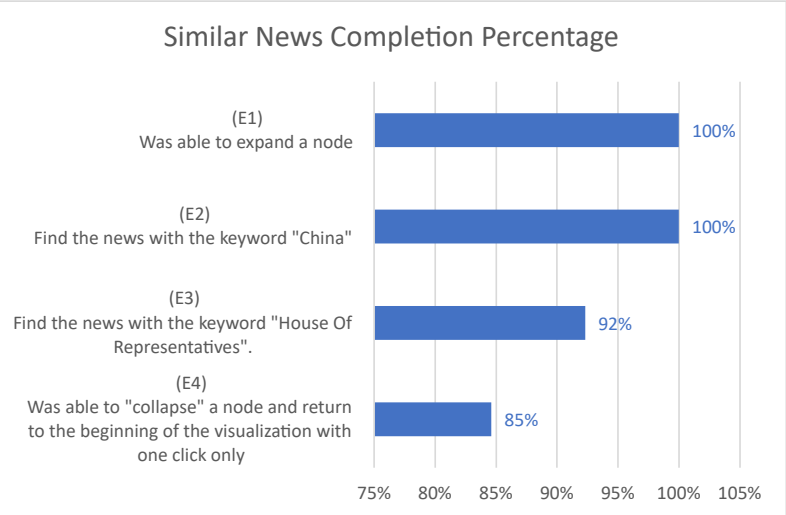
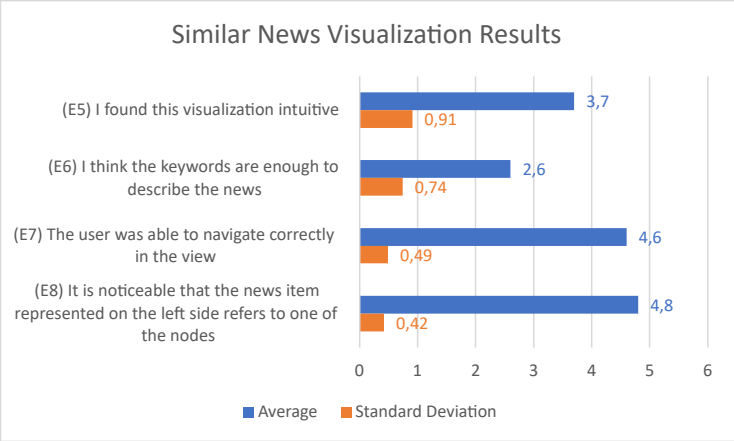
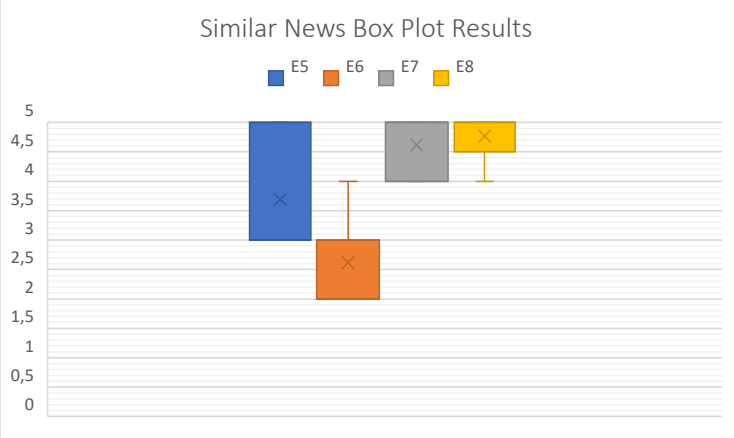


Figure 5.11: Completion percentage of E1 to E4 tasks



(a) Similar News tasks likert scale results



(b) Similar News tasks box plot results

Figure 5.12: Embedding task results (E5 to E8)

of 2.

These results point in the direction that users in general feel that just using only keywords above the nodes is not enough to describe the news articles and for this reason, it is difficult to understand what the news article will specifically address. One of the solutions could be to show more information of the news article, however, the display space is limited and it will be difficult to show a large amount of information without problems arising and affecting the visualization in a negative way, such as visual cluttering. Regarding the E7 question, the average score was of 4.6/5 with a standard deviation of 0.74. The box plot of the E7 question also exhibits that all the participants score ranges from 4 to 5. With these results it is possible to understand that the participants can correctly navigate in the visualization, which goes in line to what was expected as the visualization is simplistic. Lastly, question E8 obtained an average of 4.8 and a standard deviation of 0.42, its interquartile group ranges from 4.5 to 5 with a lower whiskers ranging from 4 to 4.5. The results indicate that users are able to understand that a node represents a news article, although it was observed that 30% of the users did not immediately notice that a node

represents a news article and when asked about that lack of immediate understanding users said that they were too focused in the visualization to notice the connection.

5.8 F - Advanced Tasks: Embeddings Visualization Testing

The last tested tab was the Embeddings tab. As with others tests, the user was first asked to complete a set of tasks by interacting with the visualization first and some questions were later asked with the purpose of obtaining more information about the visualization, its intuitiveness, the amount of information shown, the user's perception about the interacted components and what they think it represents.

Contrarily to the Similar News visualization, the Embeddings visualization contains the most interactive components out of all the visualizations present in the system and it can also be seen as a novel type of visualization which can have an affect when conducting users studies, more specifically in the interaction part of the test. The 5 tasks that the user was asked to complete can be seen in Figure 5.13. 84% of the users were able to complete the first task presented to the users. This task was manly to test if users were able to correctly navigate through the visualization. It was noticed that the inability to search for the largest sphere was not due to lack of understanding of what the largest sphere was but due to the fact that the user did not know or understood how to navigate through the visualization, which is done by dragging it with the mouse. The results show that in general, users are able to navigate through the visualization and understand the components presented in them. Although there was also some users that did not complete the tasks it was due to lack of knowledge in how to navigate and not due to some problem in the visual aspect of the visualization.

The results of the second task show that once users are familiar with visualization navigation they have no trouble completing a task similar to the firs one as 100% were able to complete the task. All the other 3 tasks also have a percentage of completion

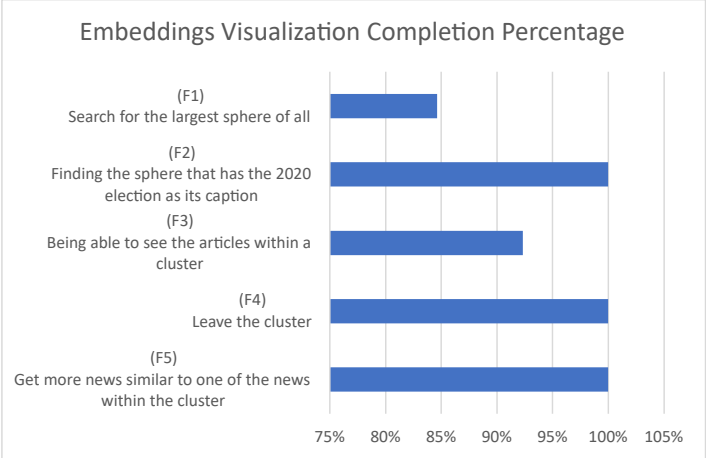
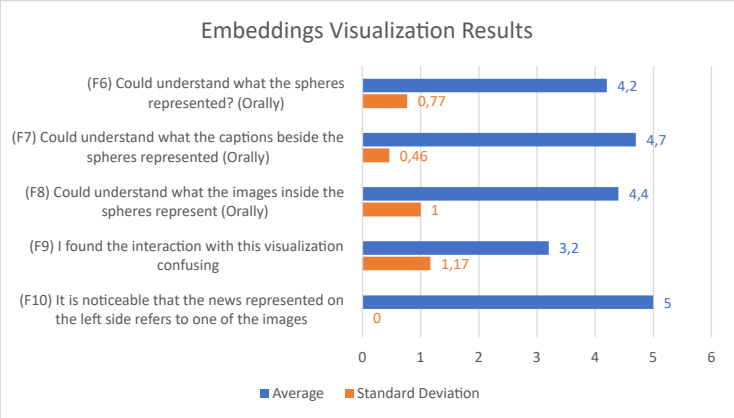
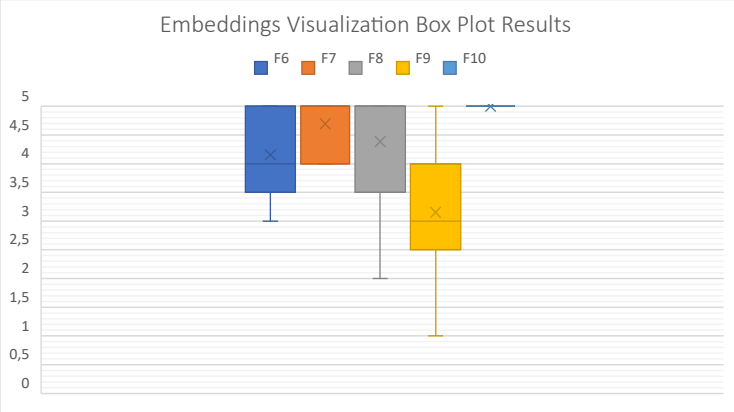


Figure 5.13: Completion percentage of F1 to F5 tasks



(a) Embeddings tasks likert scale results



(b) Embeddings tasks box plot results

Figure 5.14: Embeddings task results (F6 to F10)

between of 92% and 100% demonstrating that users have reasonable facility interacting with the visualization. Although the test is compromised of only 5 tasks it encompasses all major and expected tasks to be performed by the users while interacting with this visualization. In conclusion the tasks results are quite positive, the visualization contains some "out of the ordinary" elements in it which could could influence the user's ease of interaction with the visualization but the opposite actually happened. The only difficulty observed during the completion of the tasks was the previously reported struggle in understanding how to navigate through the visualization, struggle that disappears after users are shown how to navigate.

With regards to the questionnaire filled out by users once the tasks were completed, its questions and results are presented in Figure 5.14. The first question of the questionnaire obtained results of 4.2/5 with regards to its average and of 0.77 with regards to its standard deviation, it also obtained a median of 4, interquartile group ranging from 3.5 to 5 and a lower whiskers ranging from 3 to 3.5, showing that users understood that the spheres of the Embeddings visualization represents cluster of images. The same can be said about question F7 and F8, which also have the same purpose of understanding

if the users perception of the elements is correct. The F7 question obtained an average of 4.7 with a standard deviation of 0.46, also, all the users scores are between 4 and 5. Question F8 obtained slightly less favorable results but still positive results, the mean is similar at 4.4 altought the standard deviation is slighly higher at 1.0. Regardings the box plot, its interquartile group ranges from 5 to 3.5 with a lower whiskers ranging from 2 to 3.5. The results of the presented questions F7, F8 and F9 all have similar results and as previously mentioned, they all have the purpose of understanding if the users perception of the elements is correct and overall users did have a good understanding of what the elements represent.

With respects to the results of question F9, it obtained a score of 3.2/5 with a standard deviation of 1.17. Regarding its box plot, its interquartile group ranges from 2.5 to 4 with a median of 3, a lower and upper whiskers ranging from 1 to 2.5 and 4 to 5 respectively. These results reveal that users see this visualization as being a confusing one, some users reported that they found this visualization to be "*Out of the ordinary*" and that they "*Have never seen a visualization like this*". It is possible that one of the reasons which led users to find this visualization somewhat confusing comes from the fact that this visualization brings novelty. However, it was observable during the users tasks by their interaction with the system and the questions users made during the tasks that they did find the visualization somewhat confusing. Lastly, F10 obtained a score of 5/5 which is somewhat expected, since the way of representing the news is similar to all other visualizations tried so far.

5.9 Analysis and Discussion

In order to further analyze the system, additional questions were made to the users. One of the points analyzed is the amount of information that users perceived in relation to the visualizations so that it is possible to understand if there are any reasons to suspect that the visualizations suffer from visual cluttering, which can negatively affect those same visualizations. This component is analyzed in section 5.9.1.

Another interesting point of analysis is to understand which visualization tab users liked the most (Time Analysis, Similar News, Embeddings). Users were asked to rate each visualization tab based on their preference, which can give an idea and indicate which visualization is more likely to be "accepted" and used. Section 5.9.2 addresses this subject.

At last, the usability component of the system was tested through the use of the SUS or System Usability Scale questionnaire. A consequent analysis of the SUS score was made with the purpose of understanding were the system can be improved. Section 5.9.3 addresses this subject.

5.9.1 Visual Information Perceival

One of the main objectives of the tests was to obtain the users' perception of the information shown in the visualizations, since one of the big problems with visualizations lies in complicating and embellishing visualizations of essentially very simple data sets to the point of obscuring the essential information they were designed to convey [64]. Another problem lies in the amount of information that one tries to show in the visualizations. According to Danielle Albers Szafir [69], modern datasets may have too much data to visualize and trying to show all available data can lead to visual clutter. By having too much visual information it is not possible to find the data that matters. In order to understand if the system as a whole suffers from the previously described problems, users were asked what they think about the amount of information presented in the visualizations as a whole and four answer options were given, namely:

- Little and inadequate
- Little and adequate
- Inadequate
- Adequate
- Too much information

The results of this question are presented in Figure 5.15. The figure shows that only one of the users, which translates to a percentage of 8%, regard the amount of information shown in the visualizations as too much information. Two of the users, which translates to 15% of the participants, thought that the amount of information was little but adequate. Finally, 10 users or 77% of the participants found the amount of information to be adequate.

The results can be seen with a greatly positive connotation as 92% (77% + 15%) found the amount of information to be adequate or little and adequate, which are positive

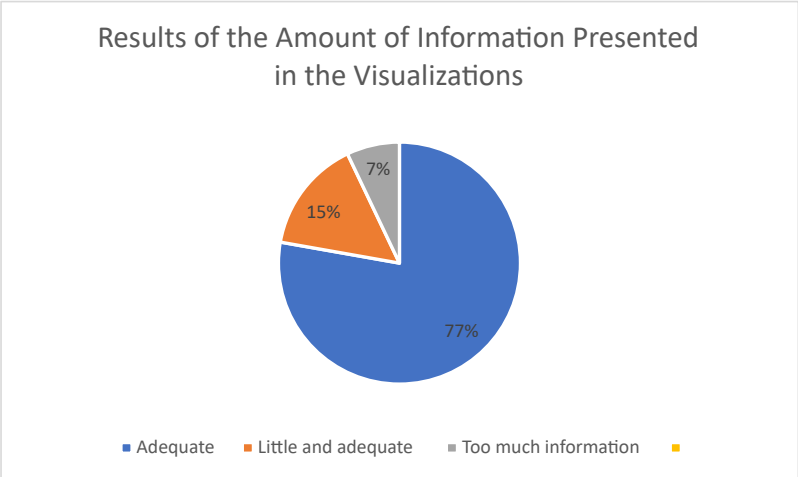


Figure 5.15: Results regarding the amount of information presented in the visualizations

answers. The fact that 15% of the participants found the amount of information to be "little" is actually not a problem but rather a positive aspect. The problem in visualizations often lays in trying not to show little information but rather too much information, which ends up negatively affecting the amount of perceived information and its learnability [64].

The results show that users consider that there is a good balance between the amount of information shown and the amount of information perceived, which also indicates that the tested visualizations do not appear to clearly suffer from cluttering or unnecessary complexity, both of which negatively affect the amount of perceived information, its learnability and the probability of larger errors of interpretation [64].

5.9.2 Visualization Tab Preference

Users were also asked to rate each visualization tab (Time Analysis, Similar News, Embeddings) based on their preference, from most liked to least liked, as this can give an idea of which type of visualization components are most liked and also which visualization are more likely to be "accepted" and used.

In order to understand which visualizations users like the most a score was calculated based on their answers, the most liked visualization would obtain a score of 3, the second most liked would obtain a score of 2, while the least liked visualization would obtain a score of 1. The results of this question can be seen in Figure 5.16. The results show that there is clearly a visualization that is preferred by all users. The Time Analysis tab was the preferred visualization tab of 100% of the participants and therefore obtained a score of 39. The Similar News tab obtained a score of 23 while the Embeddings tab obtained a score of 16. During the user's testing it was noticeable that users felt more comfortable when interacting with the visualizations in the Time Analysis tab than with other visualizations tabs. It was also noticeable its learnability rate as users almost intuitively knew how to interact with the visualizations present in the tab. The previously described observations can help understand why users liked the Time Analysis tab to the degree which they did, although a deeper analysis would be needed to understand the concrete reasons with a higher degree of certainity. Although a user likes a visualization, it does not immediately mean that the visualization is effective but it may give good indications about them. Visualizations seen as positive by users can increase the likelihood that the user will want to use that specific visualization over others and at the same time give insight into the possible details of that specific visualization that makes the user prefer it and if possible reuse certain details in other views in order to improve them.

With respects to the Similar News tab and the Embeddings tab, they placed second and third respectively. Interestingly, the visualizations are positioned according to the level of interaction difficulty observed during user testing. If the Time Analysis was the tab where less difficulty of interaction and intuitiveness was observed, it was observed that the other tabs obtained a higher degree of difficulty in interaction and intuitiveness. Similarly, the same happened between the Similar News tab and the Embeddings tab.

Although there are other factors that may have influenced the choice of users at the time of ranking the visualizations by preference, factors that would have to be further studied, it is interesting to note that the preference of users is in line with the difficulties presented in relation to interaction and intuitiveness of the different tabs.

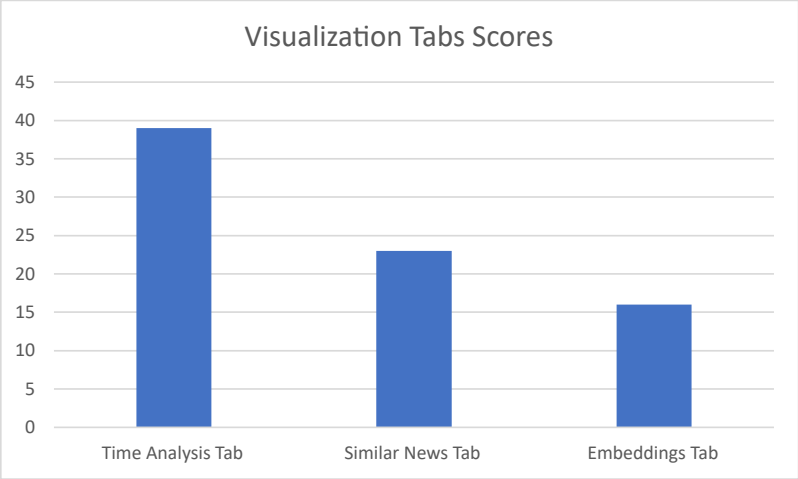


Figure 5.16: Scores of each visualization tab

5.9.3 System Usability Scale

Since the questionnaire filled out by users also contained the SUS questionnaire, in order to evaluate the results of the SUS questionnaire, a set of values and rules were followed and consequently applied to the answers obtained in the questionnaire. These sets of values and rules were defined by John Brooke [11]. The result of the application of these rules consists of a numerical value. This value represents the usability of the tested system, however, the conclusions regarding the obtained value are established by comparing the resulting value with the average numerical result validated by the results of usability tests of other systems that also made use of the SUS questionnaire, thus it is possible to evaluate and deduce where the usability of the system under test is situated. In other words, in order to understand where the usability of this system is placed it is necessary to compare the obtained result with the results of other systems.

To be able to calculate the SUS score each response is assigned a value for the SUS score calculation. The points breakdown for the responses are the following:

- Strongly Disagree: 1 point
- Disagree: 2 points
- Neutral: 3 points
- Agree: 4 points
- Strongly Agree: 5 points

Afterwards, add up the total score for all odd-numbered questions, which also corresponds to questions with a positive connotation (Table 5.1 shows this) then subtract 5 from the total (corresponding to an A score). Even-numbered questions, which inversely to positive questions, corresponds to questions with negative connotation (Table 5.1) are added up to a total score and then 25 is subtracted from the total score (corresponding to a B score). The final SUS result is calculated by multiplying the sum of the response values (A + B) by 2.5, resulting in a value comprised in the range 0-100. The value is not a percentage score but a total score out of 100.

According to Aaron Bangor et al. [5] the SUS score alone should not be used in isolation when judging the usability of a system. Factors such as the success rate and the nature of the failures observed during the system testing should play a large part when determining how usable a product is. Nevertheless, Aaron Bangor states that systems that have a SUS score 70 and above are at least passable, with better products scoring in the high 70s to upper 80s. Truly superior systems have to score better than 90. Systems that have scores of less than 70 should be considered by the author as candidates for increased scrutiny and continued improvement and should be judged to be marginal at best. The authors synthetized and compared the mean of SUS scores by quartile, adjective ratings and the acceptability of the overall SUS score (Figure 5.17). As it can be seen in the figure, a SUS score of less than 50 is evaluated as unacceptable. Scores between 50 and 70 are considered marginal while systems with 70 or more are considered acceptable.

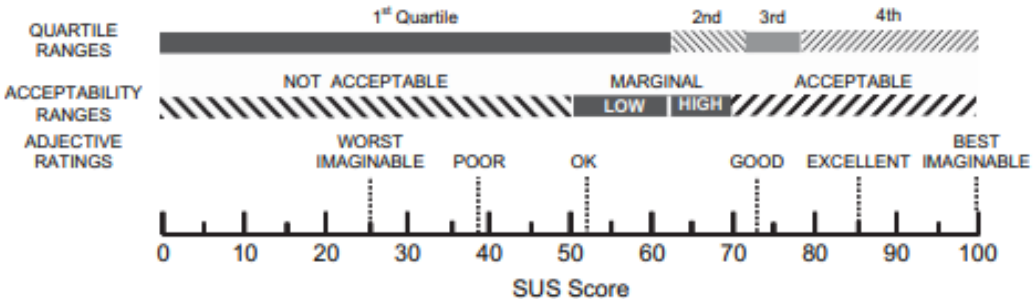


Figure 5.17: A comparison of mean System Usability Scale (SUS) scores by quartile, adjective ratings, and the acceptability of the overall SUS score. [5]

The system in question obtained a score of 77,5, which according to Bangor et al. considers the usability component of the system to be in the intermediate level between good and excellent, belonging to the 3rd quartile group (Figure 5.17). Although there is margin for improvement, the results can be seen as positive. Some of the SUS questions that "lowered" the overall SUS score and therefore the aspects that those questions address can be tackled first if possible in order to improve the usability of the system were the 2, 4 and 8. The questions are represented in table 5.1 and are as follows: 2 - *"I found the feature unnecessarily complex."*, 4 - *"I think that I would need the support of a technical person to be able to use this feature."* and 8 - *" I found the feature very cumbersome to use."*. The described questions seem to have some degree of similarity between them and it is

possible that by reducing the complexity of the system, which is the attribute mentioned in question 2, improve in turn the results of question 4 and 8, since a high complexity of the system can make the user think that he needs the help of a technical person or find the system to be more cumbersome.

6

Conclusions And Future Work

6.1 Conclusions

The resulting system from the development process of this dissertation integrates all the objectives and requirements that were initially defined in Chapter 1 and Chapter 3. In this sense, the creaction of a web application was achieved and interacts with an Embeddings System that during a user session has the purpose of recommending news pieces based on the users' search queries. This Embeddings System allows for multimodal querying which the web application supports. The developed interface also supports all the types of visualizations mentioned in Chapter 3, namely: the graph visualization, the embeddings visualization and the time analysis visualization and other types of visualizations. These visualizations are able to provide a way of viewing the users' queries in order to gain insight into the various topics that the user intends to explore during his session. The visualizations also allow the user to narrow the amount of information (i.e., news articles) shown, which might help the user find relevant news articles or proceed to another turn, with each turn being a more concrete and precise search query, leading to more refined results.

The implementation of other components that communicate with this web application was also achieved, namely a JavaScript server that is able to process and transform data to then support the visualizations with the type of data and structure they need to be able to be loaded.

A Python server was also implemented. Its implementation was not initially planned but was necessary due to the fact that the algorithms used are optimized for the Python language and not the JavaScript language. This server has, similarly to the JavaScript server, data transformation and processing functions, namely for the visualization of the embeddings that needs to make use of dimensionality reduction and clustering algorithms that are optimized for the Python language.

Finally, user tests were carried for the purpose of understanding how users interact with the system and evaluate its usability and effectiveness at exploring and finding the desired news articles. Users tested different visualizations for different ways of exploring

news articles. The Time Analysis visualization was used in order to filter the amount of news by time and common words of the returned news articles. With the Time Analysis visualization it is possible to, for example, understand how a specific topic unfolds through time by filtering the amount of news by time. The Similar News visualization allowed users to obtain similar news with respects to one of the user's interest through the use of graphs, with each node representing a similar news article. Lastly, the Embeddings visualization was used with the purpose of clustering news that are close to each other regarding their content so it is possible to separate a search topic into sub-topics that a user might be interested in. Users did not feel that the visualizations suffered from visual cluttering, as supported by the results of section 5.9.1, which shows that about 92% of the users perceived the amount of information shown to be adequate or little and adequate.

6.2 Future Work

Having concluded the implementation of the different components proposed for this dissertation, some issues to be improved in the system were identified.

Regarding the Embeddings visualization, after its implementation and consequent preliminary tests, we identified an excessive RAM memory usage by the visualization, when a large amount of images were presented. In order to circumvent the issue, three levels of image quality were used, changed at specific zoom scales. Although this approach considerably reduced the amount of memory used by the visualization, the issue persists when it has to load hundreds of images. The exploration of different memory usage reduction techniques is thus identified as a significant future objective.

An interesting feature to explore in the Embeddings visualisation would be to allow the user to split a large cluster into many smaller clusters and refresh the visualisation based on that cluster only. This would be similar to redoing the visualisation using a smaller set of news articles, consisting in the articles within the cluster that the user is interested in splitting into smaller clusters. By splitting a larger cluster into smaller clusters, it would be possible to separate the amount of news in an even more specific way, with smaller clusters having more similar and specific news articles.

Regarding the implementation of the Similar News Visualization, initially we did not foresee that the visualization would be loaded only when a user clicked on one of the news images presented in the news feed. This need arose due to the fact that the Similar News Visualization retrieves a total of 125 news articles, with each having 5 similar news articles at a depth of 3 (5*5*5), requiring a very heavy search that easily overloads the Python server, much heavier than simply returning 125 similar news of a single article. There is a need to improve this search so that the server is not overloaded with few requests.

Regarding the visualization itself, it can also be improved. The visualization's nodes (that represent news) are only described by their keywords, which although can give the user an idea of what the news article's content is about, it has the ability to hold more

information, in order to better summarize the content of a news article and give the user a more effective way of visualizing the news' content. Further research is recommended in order to understand the possibilities of improving the representation of the main characteristics of a news article on a smaller space, as the nodes used in the visualization in a summarized way that does not harm the visualization but helps it.

Lastly, the data associated with news articles contains the country that the article addresses, directly or indirectly. By having data that makes reference to the country that the article addresses, it is possible to implement visualizations that take into account the geolocation of the news article, which would allow users to filter the news articles in another way, through a new visualization. A prototype with the purpose of exploring news geolocation using the Three.js library was implemented, although not concluded. Figure 6.1 displays the visualization's prototype. It comprises a 3D world representation, with each country having a red sphere representing the amount of news, the bigger the circle the more news located in that country.

(a) 3D world visualization

(b) Zoomed 3D world visualization

Figure 6.1: Prototype of a visualization that takes into account the country that a news article addresses using a 3D earth representation

Bibliography

[1] URL: https://reactjs.org (visited on 2022-03) (cit. on p. 32).

[2] URL: https://threejs.org/ (visited on 2022-03) (cit. on p. 33).

[3] URL: https://expressjs.com/ (visited on 2022-03) (cit. on p. 33).

[4] "Exploring Large Digital Library Collections Using a Map-Based Visualisation". English. In: *Research and Advanced Technology for Digital Libraries*. Ed. by T. Aalberg, C. Papatheodorou, M. Dobreva, G. Tsakonas, and C. Farrugia. Vol. 8092. Lecture Notes in Computer Science. springer, 2013, pp. 216–227. ISBN: 9783642405006. DOI: 10.1007/978-3-642-40501-3_21 (cit. on pp. 7, 8).

[5] Bangor, Aaron, P. Kortum, P. T., Miller, and J. T. "The System Usability Scale (SUS): an Empirical evaluation". In: *International Journal of Human-Computer Interaction* 24 (2008-08), pp. 574–. DOI: 10.1080/10447310802205776 (cit. on p. 97).

[6] M. Berger, K. McDonough, and L. M. Seversky. "cite2vec: Citation-Driven Document Exploration via Word Embeddings". In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 691–700. DOI: 10.1109/TVCG.2016.2598667 (cit. on p. 20).

[7] J. Bertin. "Graphics and Graphic Information Processing". In: *Readings in Information Visualization: Using Vision to Think*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 62–65. ISBN: 1558605339 (cit. on p. 24).

[8] N. Bikakis. "Big Data Visualization Tools". In: 2018-01 (cit. on p. 13).

[9] A. Boggust, B. Carter, and A. Satyanarayan. "Embedding Comparator: Visualizing Differences in Global Structure and Local Neighborhoods via Small Multiples". In: *CoRR* abs/1912.04853 (2019). arXiv: 1912.04853. URL: http://arxiv.org/abs/1912.04853 (cit. on p. 16).

[10] M. Bostock, V. Ogievetsky, and J. Heer. "D3: Data-Driven Documents". In: *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2011). URL: http://vis.stanford.edu/papers/d3 (cit. on pp. 9, 33).

[11] J. Brooke. "SUS: A quick and dirty usability scale". In: *Usability Eval. Ind.* 189 (1995-11) (cit. on pp. 76, 96).

[12] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt. "A Text Feature Based Automatic Keyword Extraction Method for Single Documents". In: *Advances in Information Retrieval*. Ed. by G. Pasi, B. Piwowarski, L. Azzopardi, and A. Hanbury. Cham: Springer International Publishing, 2018, pp. 684–691. ISBN: 978-3-319-76941-7 (cit. on p. 10).

[13] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, X. Tong, and H. Qu. "TextFlow: Towards Better Understanding of Evolving Topics in Text". In: *IEEE transactions on visualization and computer graphics* 17 (2011-12), pp. 2412–21. DOI: 10.1109/TVCG.2011.239 (cit. on p. 6).

[14] W. Cui, Y. Wu, S. Liu, F. Wei, M. Zhou, and H. Qu. "Context-Preserving, Dynamic Word Cloud Visualization". In: *IEEE Computer Graphics and Applications* 30.6 (2010), pp. 42–53. DOI: 10.1109/MCG.2010.102 (cit. on p. 26).

[15] T. Dang, H. Nguyen, and P. Vung. "WordStream: Interactive Visualization for Topic Evolution". In: 2019-05. DOI: 10.2312/evs.20191178 (cit. on p. 26).

[16] D. Demashov and I. B. Gosudarev. "Efficiency Evaluation of Node.js Web-Server Frameworks". In: *MICSECS*. 2019 (cit. on p. 33).

[17] C. Depaolo and K. Wilkinson. "Get Your Head into the Clouds: Using Word Clouds for Analyzing Qualitative Assessment Data". In: *TechTrends* 58 (2014-05), pp. 38–44. DOI: 10.1007/s11528-014-0750-9 (cit. on p. 26).

[18] T. Devezas, J. Devezas, and S. Nunes. "Exploring a Large News Collection Using Visualization Tools". In: *Proceedings of the First International Workshop on Recent Trends in News Information Retrieval co-located with 38th European Conference on Information Retrieval (ECIR 2016), Padua, Italy, March 20, 2016*. Ed. by M. Martinez-alvarez, U. Kruschwitz, G. Kazai, F. Hopfgartner, D. Corney, R. Campos, and D. Albakour. Vol. 1568. CEUR Workshop Proceedings. Citations: dblp, scopus. 2016, 48–53 (cit. on pp. 9–11).

[19] T. Devezas, S. Nunes, and M. T. Rodríguez. "MediaViz: An Interactive Visualization Platform for Online Media Studies". In: *Proceedings of the 2015 International Workshop on Human-centric Independent Computing* (2015) (cit. on p. 9).

[20] A. Diaz-Papkovich, L. Anderson-Trocmé, and S. Gravel. "A review of UMAP in population genetics". In: *Journal of Human Genetics* 66 (2020-10), pp. 1–7. DOI: 10.1038/s10038-020-00851-4 (cit. on p. 18).

[21] A. Dillon. "The Evaluation of software usability". In: 2001 (cit. on p. 76).

[22] M. Dontcheva, M. Agrawala, and M. Cohen. "Metadata visualization for image browsing". In: *18th annual ACM symposium on user interface software and technology*. 2005 (cit. on pp. 21, 24).

[23] G. Dwyer, S. Aggarwal, and J. Stouffer. *Flask: Building Python Web Services*. Packt Publishing, 2017. ISBN: 1787288226 (cit. on p. 34).

[24] J. Etzold, A. Brousseau, P. Grimm, and T. Steiner. "Context-aware querying for multimodal search engines". In: *Proceedings of the 18th international conference on Advances in Multimedia Modeling*. Berlin, Heidelberg, 2012, pp. 728–739 (cit. on p. 2).

[25] M. Ghoniem, D. Luo, J. Yang, and W. Ribarsky. "NewsLab: Exploratory Broadcast News Video Analysis". In: 2007-10, pp. 123–130. ISBN: 978-1-4244-1659-2. DOI: 10.1109/VAST.2007.4389005 (cit. on pp. 6, 25).

[26] M. Grinberg. *Flask Web Development: Developing Web Applications with Python*. 1st. O'Reilly Media, Inc., 2014. ISBN: 1449372627 (cit. on p. 34).

[27] X. Han, Z. Wu, P. X. Huang, X. Zhang, M. Zhu, Y. Li, Y. Zhao, and L. Davis. "Automatic Spatially-Aware Fashion Concept Discovery". In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 1472–1480 (cit. on p. 20).

[28] S. Havre, B. Hetzler, and L. Nowell. "ThemeRiver: Visualizing Theme Changes over Time". In: *Proceedings of the IEEE Symposium on Information Vizualization 2000*. INFOVIS '00. USA: IEEE Computer Society, 2000, p. 115. ISBN: 0769508049 (cit. on pp. 6, 25).

[29] J. Heer and D. Boyd. "Vizster: visualizing online social networks". In: *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* (2005), pp. 32–39 (cit. on pp. 15, 17).

[30] F. Heimerl, S. Lohmann, S. Lange, and T. Ertl. "Word Cloud Explorer: Text Analytics Based on Word Clouds". In: *2014 47th Hawaii International Conference on System Sciences*. 2014, pp. 1833–1842. DOI: 10.1109/HICSS.2014.231 (cit. on p. 26).

[31] D. Herrmannova and P. Knoth. "Visual Search for Supporting Content Exploration in Large Document Collections". In: *D Lib Mag.* 18 (2012) (cit. on p. 5).

[32] C. Hirsch, J. Hosking, and J. Grundy. "Interactive Visualization Tools for Exploring the Semantic Graph of Large Knowledge Spaces". In: (2010-05) (cit. on p. 13).

[33] M. Hu, K. Wongsuphasawat, and J. Stasko. "Visualizing Social Media Content with SentenTree". In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017), pp. 621–630. DOI: 10.1109/TVCG.2016.2598590 (cit. on p. 26).

[34] M. Huang, T.-H. Huang, and J. Zhang. "TreemapBar: Visualizing Additional Dimensions of Data in Bar Chart". In: 2009-07, pp. 98–103. DOI: 10.1109/IV.2009.22 (cit. on p. 66).

[35] C. Ignat, B. Pouliquen, R. Steinberger, and T. Erjavec. "A tool set for the quick and efficient exploration of large document collections". In: (2006-10) (cit. on p. 5).

[36] A. Jaffe, M. Naaman, T. Tassa, and M. Davis. "Generating summaries and visualization for large collections of geo-referenced photographs". In: 2006-01, pp. 89–98. DOI: 10.1145/1178677.1178692 (cit. on p. 25).

[37] P. Janecek and P. Pu. "Searching with semantics: an interactive visualization technique for exploring an annotated image collection". In: *On The Move to Meaningful Internet Systems 2003: OTM 2003 Workshops*. Lecture Notes in Computer Science. 2889 (2003). Human Comput. Interaction Group, Swiss Fed. Inst. of Technol., Lausanne, Switzerland, pp. 185–196. DOI: 10.1007/978-3-540-39962-9_30. URL: http://infoscience.epfl.ch/record/115263 (cit. on pp. 21, 22, 24).

[38] J. Jcamargo Mendoza. "A strategy for interactive exploration of multimodal image collections". PhD thesis. 2014 (cit. on p. 18).

[39] M. John, E. Marbach, S. Lohmann, F. Heimerl, and T. Ertl. "MultiCloud: Interactive Word Cloud Visualization for the Analysis of Multiple Texts". In: *Proceedings of the 44th Graphics Interface Conference*. GI '18. Toronto, Canada: Canadian Human-Computer Communications Society, 2018, pp. 34–41. ISBN: 9780994786838. DOI: 10.20380/GI2018.06. URL: https://doi.org/10.20380/GI2018.06 (cit. on p. 26).

[40] I. Jolliffe. "Principal component analysis: A beginner's guide - I. Introduction and application". In: *Weather* 45 (1990-10), pp. 375–382. DOI: 10.1002/j.1477-8696.1990.tb05558.x (cit. on p. 18).

[41] S. Kaski and J. Peltonen. "Dimensionality Reduction for Data Visualization [Applications Corner]". In: *IEEE Signal Processing Magazine* 28.2 (2011), pp. 100–104. DOI: 10.1109/MSP.2010.940003 (cit. on p. 16).

[42] N. Katricheva, A. Yaskevich, A. Lisitsina, T. Zhordaniya, A. Kutuzov, E. Kuzmenko, and N. Katricheva. "Vec2graph: A Python Library for Visualizing Word Embeddings as Graphs". In: 2020-02, pp. 190–198. ISBN: 978-3-030-39574-2. DOI: 10.1007/978-3-030-39575-9_20 (cit. on p. 20).

[43] D. Keim, M. Hao, U. Dayal, and M. Hsu. "Pixel Bar Charts : A Visualization Technique for Very Large Multi-Attribute Data Sets". In: *First publ. in: Information visualization 2 (2002), 1, pp. 20-34* 1 (2002-03). DOI: 10.1057/palgrave.ivs.9500003 (cit. on p. 66).

[44] E. Kim, S. Antani, X. Huang, L. Long, and D. Demner-Fushman. "Using Relevant Regions in Image Search and Query Refinement for Medical CBIR". In: 7967 (2011-03). DOI: 10.1117/12.878192 (cit. on pp. 21, 22, 24).

[45] M. Krstajić, E. Bertini, F. Mansmann, and D. Keim. "Visual analysis of news streams with article threads". In: (2010-01), pp. 39–46. DOI: 10.1145/1833280.1833286 (cit. on pp. 8, 9, 11).

[46] S. Lehmann, U. Schwanecke, and R. Dörner. "Interactive visualization for opportunistic exploration of large document collections". In: *Information Systems* 35.2 (2010). Special Section: Context-Oriented Information Integration, pp. 260–269. ISSN: 0306-4379. DOI: https://doi.org/10.1016/j.is.2009.10.004. URL: https://www.sciencedirect.com/science/article/pii/S0306437909001021 (cit. on pp. 12, 13).

[47] J. Lewis and J. Sauro. "Revisiting the Factor Structure of the System Usability Scale". In: *Journal of Usability Studies* 12 (2017-08), pp. 183–192 (cit. on p. 77).

[48] *Likert scale*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html (visited on 2022-06) (cit. on p. 60).

[49] *Likert scale*. URL: https://www.britannica.com/topic/Likert-Scale (visited on 2022-06) (cit. on p. 79).

[50] S. Liu, M. X. Zhou, S. Pan, Y. Song, W. Qian, W. Cai, and X. Lian. "TIARA: Interactive, Topic-Based Visual Text Summarization and Analysis". In: *ACM Trans. Intell. Syst. Technol.* 3.2 (2012-02). ISSN: 2157-6904. DOI: 10.1145/2089094.2089101. URL: https://doi.org/10.1145/2089094.2089101 (cit. on pp. 24, 25).

[51] S. Lohmann, F. Heimerl, F. Bopp, M. Burch, and T. Ertl. "Concentri Cloud: Word Cloud Visualization for Multiple Text Documents". In: 2015-07, pp. 114–120. DOI: 10.1109/iV.2015.30 (cit. on p. 26).

[52] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/master/template.pdf (cit. on p. ii).

[53] L. van der Maaten and G. Hinton. "Visualizing data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008-11), pp. 2579–2605 (cit. on p. 18).

[54] L. McInnes, J. Healy, N. Saul, and L. Grossberger. "UMAP: Uniform Manifold Approximation and Projection". In: *The Journal of Open Source Software* 3.29 (2018), p. 861 (cit. on p. 18).

[55] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems* 26 (2013-10) (cit. on p. 16).

[56] F. Miley and A. Read. "Using word clouds to develop proactive learners". In: *Journal of the Scholarship of Teaching and Learning* 11 (2011), pp. 91–110 (cit. on p. 26).

[57] D. Paranyushkin. "InfraNodus: Generating Insight Using Text Network Analysis". In: *The World Wide Web Conference*. WWW '19. San Francisco, CA, USA: Association for Computing Machinery, 2019, pp. 3584–3589. ISBN: 9781450366748. DOI: 10.1145/3308558.3314123. URL: https://doi.org/10.1145/3308558.3314123 (cit. on pp. 14, 74).

[58] A. Pasquali, V. Mangaravite, R. Campos, A. M. Jorge, and A. Jatowt. "Interactive System for Automatically Generating Temporal Narratives". In: *Advances in Information Retrieval*. Ed. by L. Azzopardi, B. Stein, N. Fuhr, P. Mayr, C. Hauff, and D. Hiemstra. Cham: Springer International Publishing, 2019, pp. 251–255. ISBN: 978-3-030-15719-7 (cit. on pp. 10, 11).

[59] C. Paul, J. Chang, A. Endert, N. Cramer, D. Gillen, S. Hampton, R. Burtner, R. Perko, and K. Cook. "TexTonic: Interactive visualization for exploration and discovery of very large text collections". In: *Information Visualization* 18 (2018-07), p. 147387161878539. DOI: 10.1177/1473871618785390 (cit. on p. 7).

[60] C. Peters. "Building Rich Internet Applications with Node. js and Express. js". In: *Rich Internet Applications w/HTML and Javascript* (2017), p. 15 (cit. on p. 33).

[61] J. Pokorny, A. Norman, A. Zanesco, S. Bauer-Wu, B. Sahdra, and C. Saron. "Network Analysis for the Visualization and Analysis of Qualitative Data". In: *Psychological Methods* 23 (2016-11). DOI: 10.1037/met0000129 (cit. on pp. 14, 15, 110, 111).

[62] N. Rahmah and I. S. Sitanggang. "Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra". In: *IOP Conference Series: Earth and Environmental Science* 31 (2016-01), p. 012012. DOI: 10.1088/1755-1315/31/1/012012. URL: https://doi.org/10.1088/1755-1315/31/1/012012 (cit. on p. 60).

[63] M. Reckziegel, M. F. Cheema, G. Scheuermann, and S. Jänicke. "Predominance Tag Maps". In: *IEEE Transactions on Visualization and Computer Graphics* 24.6 (2018), pp. 1893–1904. DOI: 10.1109/TVCG.2018.2816208 (cit. on pp. 25, 26).

[64] R. Reitsma and A. Marks. "The Future of Data: Too Much Visualization — Too Little Understanding?" In: *Dialectic* 2 (2019-2019). URL: https://quod.lib.umich.edu/d/dialectic/14932326.0002.207?view=text;rgn=main (cit. on pp. 94, 95).

[65] J. Risch. "On the role of metaphor in information visualization". In: (2008-09) (cit. on p. 5).

[66] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications". In: *Data Mining and Knowledge Discovery* 2 (2004), pp. 169–194 (cit. on p. 61).

[67] H. Siirtola, P. Isokoski, T. Säily, and T. Nevalainen. "Interactive Text Visualization with Text Variation Explorer". In: *2016 20th International Conference Information Visualisation (IV)*. 2016, pp. 330–335. DOI: 10.1109/IV.2016.57 (cit. on p. 24).

[68] D. Smilkov, N. Thorat, C. Nicholson, E. Reif, F. Viégas, and M. Wattenberg. "Embedding Projector: Interactive Visualization and Interpretation of Embeddings". In: (2016-11) (cit. on pp. 18, 19).

[69] D. A. Szafir. "The Good, the Bad, and the Biased: Five Ways Visualizations Can Mislead (and How to Fix Them)". In: *Interactions* 25.4 (2018-06), pp. 26–33. ISSN: 1072-5520. DOI: 10.1145/3231772. URL: https://doi.org/10.1145/3231772 (cit. on p. 94).

[70] W. S. Torgerson. "Multidimensional scaling: I. Theory and method". In: *Psychometrika* 17 (1952), pp. 401–419 (cit. on p. 16).

[71] A. Tran, A. Mathews, and L. Xie. "Transform and Tell: Entity-Aware News Image Captioning". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 13032–13042. DOI: 10.1109/CVPR42600.2020.01305 (cit. on pp. 2, 29).

[72] F. B. Viegas, M. Wattenberg, and J. Feinberg. "Participatory Visualization with Wordle". In: *IEEE Transactions on Visualization and Computer Graphics* 15.6 (2009), pp. 1137–1144. DOI: 10.1109/TVCG.2009.171 (cit. on p. 6).

[73] F. B. Viégas and M. Wattenberg. "TIMELINES

Tag Clouds and the Case for Vernacular Visualization". In: *Interactions* 15.4 (2008-07), pp. 49–52. ISSN: 1072-5520. DOI: 10.1145/1374489.1374501. URL: https://doi.org/10.1145/1374489.1374501 (cit. on p. 25).

[74] M. Wattenberg and F. B. Viégas. "The Word Tree, an Interactive Visual Concordance". In: *IEEE Transactions on Visualization and Computer Graphics* 14.6 (2008), pp. 1221–1228. DOI: 10.1109/TVCG.2008.172 (cit. on p. 26).

[75] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu. "OpinionFlow: Visual Analysis of Opinion Diffusion on Social Media". In: *IEEE Transactions on Visualization and Computer Graphics* 20 (2014), pp. 1763–1772 (cit. on p. 6).

[76] X. Xie, X. Cai, J. Zhou, N. Cao, and Y. Wu. "A Semantic-Based Method for Visualizing Large Image Collections". In: *IEEE Transactions on Visualization and Computer Graphics* 25.7 (2019), pp. 2362–2377. DOI: 10.1109/TVCG.2018.2835485 (cit. on pp. 22–24).

[77] J. Yang, J. Fan, D. Hubball, Y. Gao, H. Luo, W. Ribarsky, and M. Ward. "Semantic Image Browser: Bridging Information Visualization with Automated Intelligent Image Analysis". In: *2006 IEEE Symposium On Visual Analytics Science And Technology*. 2006, pp. 191–198. DOI: 10.1109/VAST.2006.261425 (cit. on pp. 21, 24).

[78] J. Yang, D. Luo, and Y. Liu. "Newdle: Interactive Visual Exploration of Large Online News Collections". In: *IEEE Computer Graphics and Applications* 30.5 (2010), pp. 32–41. DOI: 10.1109/MCG.2010.93 (cit. on pp. 6, 7, 11).

[79] A. A. Zapata. "An in-depth review of the t-sne with applications". Bachelor's Thesis. 2019 (cit. on p. 18).

[80] N. Q. Zhu. *Data Visualization with D3.Js Cookbook*. Packt Publishing, 2013. ISBN: 178216216X (cit. on p. 33).

I

ANNEX

Figure I.1: Example of a network of the system [61]

Figure I.2: Example of a network of the system [61] retirado de https://github.com/scienceai/tsne-js