TÉCNICO LISBOA

ArgumentNext

Visualizing arguments in the real world

Ana Sofia Marques Silva

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Daniel Jorge Viegas Gonçalves
Prof. Bruno Emanuel Da Graça Martins

Examination Committee

Chairperson: Prof. José Alberto Rodrigues Pereira Sardinha
Supervisor: Prof. Daniel Jorge Viegas Gonçalves
Member of the Committee: Prof. Alfredo Manuel Dos Santos Ferreira Júnior

November 2022

Acknowledgments

I am extremely grateful to my professor, Daniel Gonçalves, who supported and guided me throughout the biggest challenge of my academic life. Thank you for your availability and attention and for always pushing me to do better.

This work would not have come this far without the team of the DARGMINTS project. With your friendliness and support, I was able to guide my work to a better outcome.

Words cannot express my gratitude to my mother, Mafalda, who is there every step of the way to tell me I can do it! Thank you so much for all your love!

Thank you to my sister, Mónica, who can understand me like no one else. Thank you for having time to listen to me and for all the support!

I am also grateful to my father, Mário, who always worried if I was working too much. Thank you for supporting me throughout my life!

To my grandparents, Maria and Mário, thank you for always caring for me and making sure I had everything I needed.

To my grandmother, Bina, I am grateful for all the love and support you gave me throughout my life.

Thanks should also go to my best friends, Margarida and João, who helped me, not only during the development of the thesis, but also during my college years. I am forever grateful for all your love and care.

I would like to extend my sincere thanks to all the people that were part of my academic life, such as professors, friends and colleagues.

Finally, I could not forget my lovely cats, Kiara, Cece and Luki, who were my company and my emotional support throughout my academic life.

Abstract

Argumentation has been part of humanity's communication for centuries as a way of showing if someone is in favor or against an idea. Studies claim practicing argumentation can improve critical thinking. However, analysing argumentative text is a complex task, due to its characteristics. To address this problem, visual representations were developed, such as argument maps, which show the structure of an argument, being the most common representation. In order to understand if there were characteristics of the argument that might have influenced its creation, it is necessary to turn to other visual techniques.

As part of the DARGMINTS project, the goal of this work is to create visualizations that facilitate the comprehension of argumentative text, both by analysts and non-experts. This project, firstly, lead to the development of a framework that created visualizations for this kind of data. However, it needed to be adapted in order to use real data and contain the necessary tools for the demonstrators.

These demonstrators consist in a dashboard for the opinion articles case study, with interactive visual components that allows answering various questions about the documents, and in a storytelling visualization for the parliamentary debates domain, which uses information found through the analysis of data and visual components to tell a story.

During the evaluation of the platforms, the dashboard was confusing to non-experts, due to the concepts' complexity. Nevertheless, it was successful among experts, which were more contextualized, being able to use the platform more easily. In contrary, the storytelling visualization was well received, since the platform is more accessible, concerning its data and its structure.

Keywords

Argument visualization; Visual Analytics; Natural Language Processing; Text Visualization; Storytelling Visualization; Information Visualization.

Resumo

Argumentação faz parte da nossa comunicação há séculos para mostrar quando se está a favor ou contra uma ideia. Estudos mostram que praticar argumentação pode melhorar o pensamento crítico. No entanto, analisar texto argumentativo é uma tarefa complexa, devido às suas características. Para abordar este problema, foram criadas representações visuais, onde a mais comum são mapas de argumento, que mostram a estrutura de um argumento. Porém, para perceber se houve características do argumento que influenciaram a sua criação, é necessário olhar para outras técnicas visuais.

Como parte do projeto DARGMINTS, o objetivo deste trabalho é criar visualizações que ajudam na compreensão de texto argumentativo, por analistas e não-peritos. Este projeto, inicialmente, conduziu à criação de uma framework que cria visualizações para este tipo de dados. Porém, precisava de ser adaptada para usar dados reais e ter as ferramentas necessárias para os demonstradores.

Estes demonstradores consistem num dashboard para o caso dos artigos de opinião, com componentes visuais interativas que permitem responder a várias questões sobre os documentos, e numa visualização storytelling para o caso dos debates parlamentares, que usa informação resultante da análise dos dados e componentes visuais para contar uma história.

Durante a avaliação das plataformas, o dashboard foi confuso para os não-peritos, devido à complexidade dos conceitos. No entanto, teve sucesso entre peritos, que estavam mais contextualizados, e conseguiram navegar pela plataforma mais facilmente. Contrariamente, a visualização storytelling foi bem recebida, visto que era mais acessível, relativamente aos seus dados e à sua estrutura.

Palavras Chave

Visualização de argumentos; Análise visual; Processamento de Linguagem Natural; Visualização de texto; Visualização narrativa; Visualização de informação.

Contents

1 Introduction **1**

1.1 Motivation . . . 2

1.2 DARGMINTS Project . . . 2

1.3 Objective . . . 3

1.4 Organization of the Document . . . 4

2 State of The Art **5**

2.1 Background . . . 6

2.2 Argument Visualization . . . 7

2.3 Text Visualization . . . 13

2.3.1 Storyline visualizations . . . 19

2.4 Storytelling visualizations . . . 21

2.4.1 Storytelling approaches . . . 21

2.4.2 Examples of storytelling visualizations . . . 22

2.5 Discussion . . . 24

3 Solution **28**

3.1 Framework's Previous Version . . . 29

3.1.1 Architecture and Technology . . . 29

3.1.2 Limitations . . . 31

3.2 Requirements . . . 32

3.2.1 Opinion Articles . . . 32

3.2.2 Parliamentary Debates . . . 32

3.3 Architecture and Technology . . . 33

3.3.1 Datasets . . . 33

3.4 Visual Argument Framework . . . 34

3.4.1 Visual Components . . . 34

3.4.1.A List Component . . . 34

3.4.1.B Filter Component . . . 35

3.4.1.C Text Component . . . 36
3.4.1.D ArgumentMap Component . . . 37
3.4.1.E Network Graph Component . . . 37
3.4.1.F Distribution Component . . . 39
3.4.1.G MapOverview Component . . . 40
3.4.1.H BarChart Component . . . 40
3.4.1.I StackedBarChart Component . . . 41
3.4.1.J PositionChart Component . . . 41
3.4.1.K LineChart Component . . . 42
3.5 Dashboard . . . 43
3.5.1 First Idea . . . 43
3.5.2 First Attempt . . . 44
3.5.3 Prototype Changes . . . 46
3.5.4 Final Approach . . . 48
3.6 ScrollyTeller . . . 50
3.6.1 ScrollyTeller Structure . . . 51
3.6.2 Technology . . . 51
3.6.3 Story . . . 51
3.6.4 First Idea . . . 52
3.6.5 Final Prototype . . . 53
4 Evaluation **55**
4.1 Tests with Non-Experts . . . 56
4.1.1 Users Sample . . . 56
4.1.2 Dashboard Testing . . . 57
4.1.2.A Network Graph . . . 59
4.1.2.B Annotators . . . 60
4.1.2.C Other Problems . . . 61
4.1.2.D Special Cases . . . 61
4.1.2.E SUS . . . 61
4.1.3 ScrollyTeller Testing . . . 63
4.1.3.A Writing . . . 63
4.1.3.B Components . . . 63
4.1.3.C Overall Story . . . 64
4.1.3.D SUS . . . 64
4.2 Tests with Experts . . . 65

4.2.1 Users Sample . . . 65
4.2.2 Dashboard Testing . . . 66
4.2.3 ScrollyTeller Testing . . . 67
4.3 Discussion . . . 69
4.4 Improvements . . . 69
4.4.1 Dashboard . . . 69
4.4.2 ScrollyTeller . . . 70
5 Conclusion **72**
5.1 Limitations and Future Work . . . 74
Bibliography **74**
A Datasets **77**
A.1 Opinion Articles . . . 78
A.2 Parliamentary Debates . . . 79
B Testing Protocol – Non-Experts **80**
C Testing Protocol – Experts **90**

List of Figures

2.1 Argument map example 7
2.2 Reason!Able screenshot showing an argument map about snakes 8
2.3 Rationale screenshot showing an argument map about vaccines 9
2.4 bCisive screenshot showing an argument map about Obama administration 9
2.5 Convince me screenshot showing various visualization marks 9
2.6 SEAS screenshot showing the supporting information for a primitive question 10
2.7 MindMup screenshot showing an example of an argument map 11
2.8 OVA platform screenshot showing an argument map based on a dialogue 11
2.9 Screenshot of Argdown Sandbox 11
2.10 Screenshot of an argument in Kialo showing its pros and cons 11
2.11 Screenshot of the visual analysis system for essays. The ArguLines are on the right, the AUOT on the top left and the text view on the bottom left 12
2.12 Screenshot of MonkeyLearn demo 13
2.13 Example of Collocates Graph tool 14
2.14 Example of RezoViz tool 14
2.15 Example of TextualArc tool 14
2.16 Screenshot of ToPIN showing learners' time-anchored comments on a philosophy course 15
2.17 Visual encoding of SolarMap showing the different facets and keywords for the topic facet "Disease" 15
2.18 TimeRayMaps visualization showing two episodes in the news story of the Syrian war. Highlighted region shows that the episode of "Kurds declaring federal region in the North" is increasingly discussed by Russian and US sources. 16
2.19 The ConceptScope interface representing a research paper discussing animation. It consists of a Bubble Treemap (a) providing a conceptual overview; the transcript (b) and text (c) views; a level slicer (d) and a list presentation (e); and a tooltip (f) with more details about the concept 17

2.20 Textplorer interface showing the different text overviews on the left, the code distribution overview, the text display, the list of generated codes, the word cloud and the metadata filters . . . 18
2.21 StanceVis Prime visualization showing its multiple views . . . 19
2.22 Storyline visualization created by StoryFlow about the movie The Lord of the Rings . . . 20
2.23 StoryPrint visualization for the *500 days of Summer* movie showing the emotions of each character . . . 20
2.24 Screenshot of one of the visual components of the ”Are Pop Lyrics Getting More Repetitive?” page . . . 22
2.25 Screenshot of The Guardian’s page, ”Homan Square: A portrait of Chicago’s detainees” . . . 23

3.1 Example of an HTML element corresponding to a visual map component of the framework 30
3.2 Screenshot of the previous version of the Visual Argument Framework . . . 30
3.3 Architecture Diagram . . . 33
3.4 Example of the list component . . . 35
3.5 Example of the filter bar component . . . 35
3.6 Example of the text component with all annotators selected . . . 36
3.7 Example of the text component with one annotator selected . . . 36
3.8 Example of the argument map component . . . 37
3.9 Example of the network graph component using Cytoscape.js . . . 38
3.10 Example of the network graph component using D3.js . . . 39
3.11 Example of the distribution component using the paragraphs in the axis . . . 39
3.12 Example of the distribution component using the time in the axis . . . 39
3.13 Example of the map overview component . . . 40
3.14 Example of the bar chart component . . . 41
3.15 Example of the stacked bar chart component . . . 41
3.16 Example of the position chart component . . . 42
3.17 Example of the line chart component . . . 42
3.18 Dashboard Interface Prototype . . . 43
3.19 Screenshot of the first prototype of the Dashboard . . . 45
3.20 Screenshot of argument map in ArgMine . . . 46
3.21 Screenshot of the second prototype of the Dashboard . . . 47
3.22 Example of a D3.js force model creation . . . 47
3.23 Screenshot of the third prototype of the Dashboard . . . 48
3.24 Screenshot of the global view of the Dashboard . . . 49
3.25 Screenshot of the specific view of the Dashboard . . . 50

3.26 Screenshot of the initial page of the ScrollyTeller 52
3.27 Screenshot of the bar chart component with buttons, with option ”PEV” 53
3.28 Screenshot of the bar chart component with buttons, with option ”PSD” 53
3.29 Screenshot of the stacked bar chart component before scroll 53
3.30 Screenshot of the stacked bar chart component after scroll 53
3.31 Screenshot of the exploratory area of the ScrollyTeller 54

4.1 Age groups of the users of the first testing phase 57
4.2 Correct and incorrect answers for the charts section of the initial survey 57
4.3 Correct and incorrect answers for the Dashboard tasks 57
4.4 NASA-TLX survey scores for the Dashboard tasks in the first testing phase 58
4.5 Time spent on each task of the Dashboard 58
4.6 Dashboard’s NASA-TLX survey results for Task 2 59
4.7 Dashboard’s NASA-TLX survey results for Task 8 60
4.8 Dashboard’s SUS survey results in the first testing phase 62
4.9 Correct and incorrect answers for the ScrollyTeller tasks 62
4.10 NASA-TLX survey scores for the ScrollyTeller tasks in the first testing phase 63
4.11 ScrollyTeller Task 8 answers 64
4.12 ScrollyTeller’s SUS survey results in the first testing phase 65
4.13 NASA-TLX survey scores for the Dashboard tasks in the second testing phase 66
4.14 SUS survey results for the Dashboard in the second testing phase 66
4.15 ScrollyTeller Task 6 answers 68
4.16 SUS survey results for the ScrollyTeller in the second testing phase 68

List of Tables

2.1 Analysis types for each previously seen argument and text visualizations 25
2.2 Behaviour types for each previously seen storytelling visualization 26

Acronyms

ADU	Argumentative Discourse Unit
AM	Argument Mining
API	Application Programming Interface
AUOT	Argument Unit Occurrence Tree
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
DARGMINTS	Discourse Analysis and Argumentation Mining from Text Sources
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
JS	JavaScript
JSON	JavaScript Object Notation
NASA-TLX	NASA Task Load Index
NLP	Natural Language Processing
OECD	Organisation for Economic Co-operation and Development
POS	Parts-Of-Speech
PS	Partido Socialista
PSD	Partido Social Democrata
SEAS	Structured Evidential Argumentation System
SUS	System Usability Scale
SVG	Scalable Vector Graphic
VA	Visual Argument

1

Introduction

Contents

1.1 Motivation 2
1.2 DARGMINTS Project 2
1.3 Objective 3
1.4 Organization of the Document 4

An argument is a statement or series of statements that is used to convince people about an idea or opinion. Argumentation has been present in humanity for centuries since it is how humans learn how to show they are in favor or against a claim. This process is very important since it stimulates communication and rational thinking. Our society, specially politics, is built on argumentation: people try to convince others about their theories and others support or oppose to these by creating their own arguments.

1.1 Motivation

To better understand arguments, text is not enough. That's why that, in the nineteenth century, handwritten argument mapping was introduced [1]. In the twentieth first century, argument maps could already be created in digital tools.

Since then, multiple tools have been created for argument visualization. However, the way they convey the logical structure of an argument does not change much from one to other since most of them use argument maps similar to the ones from the beginning. It is fair to conclude that the visual representation of arguments has not changed significantly since its inception.

Contrary to this, non-argumentative text visualizations have evolved a lot throughout the years. In the beginning, text analysis was done by reading it multiple times and taking side notes. Although this technique is still used today, the increased use of computers provided means for the emerging of different digital tools to facilitate the exploration of this kind of text.

These tools help analysts find patterns as well as answer questions, for instance, about topics, stance, sentiment and time. These characteristics can also be present in argumentative text, however, there are no visualizations in the argumentation area that offer such analysis. Therefore, new visual representations are needed to facilitate the analysis of argumentative text.

Having this in consideration, this work joins argument maps with what was learned from text visualizations throughout the years, thus creating new ways of visualizing arguments that are more intuitive and easier to understand.

1.2 DARGMINTS Project

This work is part of and was partly funded by the Discourse Analysis and Argumentation Mining from Text Sources (DARGMINTS) project[1] that uses Natural Language Processing (NLP) algorithms and Argument Mining (AM) to extract arguments from texts. This project focuses on the Portuguese language, for which there is no relevant prior work, considering three main domains: opinion articles,

[1] DARGMINTS project: https://web.fe.up.pt/~dargmints/doku.php

parliamentary debates and political online forums. For the last there is no data yet.

They also envision the development of new interactive visualizations for exploring argumentation patterns and processes, in areas related with media studies, political science, forensics and linguistics.

The development of new visualizations that facilitate the analysis of argumentative text, thus, aligns with their goal. And, since, the DARGMINTS project team is mainly people from the linguistics area, they are the best people to guide this work to its best outcome. Therefore, during the development of this work, there were weekly meetings to ensure the visualizations to be created actually helped in the current analysis problem, where solutions were discussed in order to find the best visual representations to solve the present issue.

The creation of new applications for this type of analysis will, thus, be a great addiction for this project, since it will help them achieve their goal: analyse the data extracted through NLP algorithms.

1.3 Objective

Being a part of the DARGMINTS project, the goal of this thesis is to **create a visualization for each case study with new and creative visual components that facilitate the analysis of argumentative text.**

To accomplish this, a framework, created by a colleague a year ago [2], called the Visual Argument (VA) Framework, is used as a foundation for this work. This framework is a modular adaptive web platform that gathers some effective visual components for argumentative text. However, it used as case study mock-up data only about one of the domains: the parliamentary debates. Therefore, the VA Framework needed to be adapted to include real data produced by the DARGMINTS project NLP algorithms. Besides that, and although the framework allows customization, new visual components and new more intuitive and creative ways of doing the same tasks were created.

Since the case studies are different from each other, distinct demonstrators were developed, the Dashboard and the ScrollyTeller. The first platform uses the opinion articles dataset and focuses on deep argument analysis, namely their evolution throughout the document and the components that constituted their structure. The second platform consists in a storytelling visualization using data from parliamentary debates that tells a story about how initiatives get rejected according to the existence of absolute majorities.

The demonstrators were tested in a a two-phased testing: first by non-experts, people outside the platform's domain, and then by experts, people from the DARGMINTS project itself. It was possible to receive different opinions about the two platforms. The Dashboard was seen as a more complex tool since it required more knowledge of argumentation concepts, which confused non-experts, while the ScrollyTeller was perceived as a more accessible platform. The received feedback allowed for the

improvement of the demonstrators accordingly to achieve the final result, presented at the end of the document.

Since the VA Framework was able to create such distinct demonstrators, it proved to be very versatile, while gathering different opinions from the users, whether they were experts or not.

1.4 Organization of the Document

This document is divided in five chapters. Following the introduction, it is listed a series of crucial concepts for the understanding of this work, such as argument, argument mining, argument annotations and Argumentative Discourse Unit (ADU), accompanied by various different approaches that will serve as an inspiration to the developed solution, in a chapter called state of the art.

This chapter is succeeded by the solution chapter, where the previous version of the framework that served as a basis is explained and the requirements and architecture of the two demonstrators are established. This chapter then describes the development process for both the improvement of the framework and the creation of the two platforms.

Following the development of the solution, the platforms were evaluated. The testing process and the results are shown in the evaluation chapter. In the last chapter, conclusion, an overview of all the work is done, presenting the solution's limitations and the future work.

2

State of The Art

Contents

2.1 Background 6
2.2 Argument Visualization 7
2.3 Text Visualization 13
2.4 Storytelling visualizations 21
2.5 Discussion 24

To create a demonstrator for each case study, it is important to see the various types of visualizations that exist to analyse text. In this chapter, it will be presented the information that was gathered in relation to argument, text and storytelling visualizations. At first, the research focused more in visualizations that allowed the analysis of argumentative text. However, these tools did not evolve significantly along the years. Therefore, non-argumentative text visualizations were considered, which allow the analysis of more characteristics and have more variety. One type of a text visualization is a storytelling visualization, which is more interactive and instead of being exploratory, it leads the user to a specific story.

This chapter starts with a section regarding some concepts about argumentation, that helps to understand the work better. Afterwards, this chapter has three sections related to the different types of visualizations: argument, text and storytelling. In the end, it has a final section, where the visualizations are compared and discussed regarding their characteristics and techniques.

2.1 Background

To better understand the presented work, some concepts related to argumentative text need to defined and explained. These notions will help acknowledge how arguments are recognized in text and who annotates the text in order to create argument visualizations.

An argument is a set of statements that have relations of inference between themselves. One is the conclusion, presented as being true, which can be supported or attacked by other statements, called premises of that argument.

AM is the research area with the goal of the automatic identification and detection of the argumentative structure presented in a natural language text with the help of computer programs.

This process faces a lot of challenges, since natural language text can be ambiguous and present different writing styles and implicit context. However, argument mining has great potential, as it is able to inform the positions that are being expressed as well as tell the reasons why people should believe in those specific positions.

In order for a computer to automatically extract argumentative structures from natural language text, it is necessary to manually annotate the text first so the computer can learn, through machine learning techniques, how to perform this task. Manual annotation of arguments follows three steps [3]: text segmentation, argument and non-argument classification and structure of the argument.

In text segmentation, fragments of the original text are extracted which are called ADUs, that correspond to atomic units of discourse. The next step, argument and non-argument classification, consists in throwing away the parts of the segments previously extracted that are not relevant to the argument. Finally, using the segments important to the argument, it is possible to create its structure, by analysing the relations between the segments.

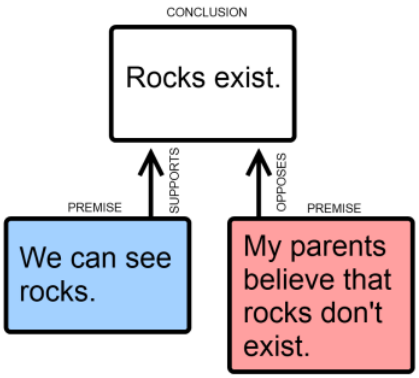

Figure 2.1: Argument map example

According to Reed and Lawrence [3], there are some defined structures for arguments: convergent, linked, divergent and sequential arguments. In convergent arguments, multiple premises independently support or attack one conclusion, while in linked arguments, the premises work together to support or attack the same conclusion. In a divergent argument, the same premise supports or attacks more that one conclusion. In sequential arguments, one premise supports another and this, in turn, supports a conclusion. These structures can then be combined in a hybrid argument structure.

As said before, an ADU is the minimal unit of analysis. This corresponds to a rectangle in the argument maps that will be explained in the state of art below. In Fig.2.1, an ADU corresponds to one of the blue, red or white rectangles of the showed map. In other words, it represents a statement in this type of argument visualization. There are five types of ADUs:

- **Fact** – where the information is presented as being true, and can be verified;
- **Value** – regarding opinions, can present position of ethical, estetic and political nature;
- **Value (+)** – when Value statements are associated with a positive polarity;
- **Value (-)** – when Value statements are associated with a negative polarity;
- **Directive** – when a certain action is invoked;

2.2 Argument Visualization

Argument visualizations have not changed substantially over the years and consist mainly in argument maps. An argument map is a diagram showing the logical structure of an argument. Typically it consists in nodes which represent propositions and links which correspond to relationships such as evidential support. Argument maps are visually similar to mind maps and concept maps [4]. Mind maps are used to organize information in order to explore associations between ideas. Concept maps, on the other hand, not only create associations between concepts but classify the type of relationship they have to

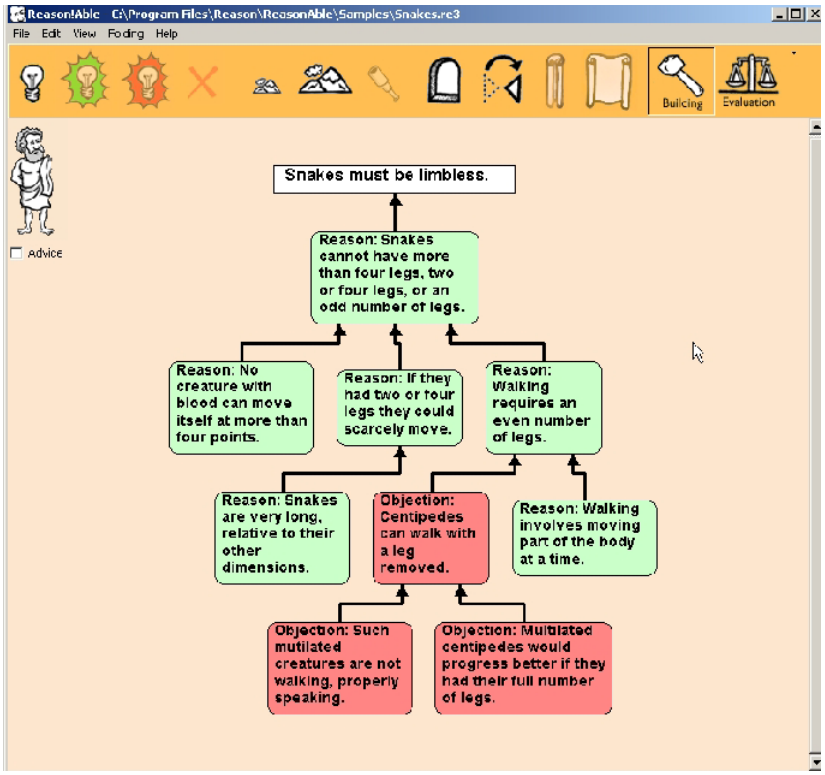

Figure 2.2: Reason!Able screenshot showing an argument map about snakes

better structure knowledge. Unlike these two mapping tools, argument mapping is not focused on the relationships between the propositions. Instead, it is more interested in the inferential basis for a claim.

Argument visualization only emerged in the nineteenth century, and it was used by Richard Whately in 1836 in a logic textbook [1]. In the 1990s, due to the evolution of computers and the development of specially designed software for this field, there was an increase in the interest in argument mapping.

In the twentieth first century, different digital tools started to emerge. One of the first was Reason!Able [5], developed in 2000. In this tool, as seen in Fig.2.2, claims, reasons, and objections are represented by rectangles with different colors for each while the relation between them is represented by arrows. Reasons and objections are complex objects that can be unfolded to show the full set of premises that are part of them. One aspect that makes Reason!Able different from other tools of this type is the fact that it has the full text in the nodes themselves instead of having them in a separate list, which makes it easier to analyse.

In 2006, the creator of Reason!Able, Tim van Gelder, developed another argument mapping tool called Rationale [6]. The main difference from the first one is that, instead of having colored rectangles for supporting and opposing claims, it groups nodes and adds a colored area around them corresponding to their type, which can be seen in Fig.2.3. These last two tools were directed to the educational field

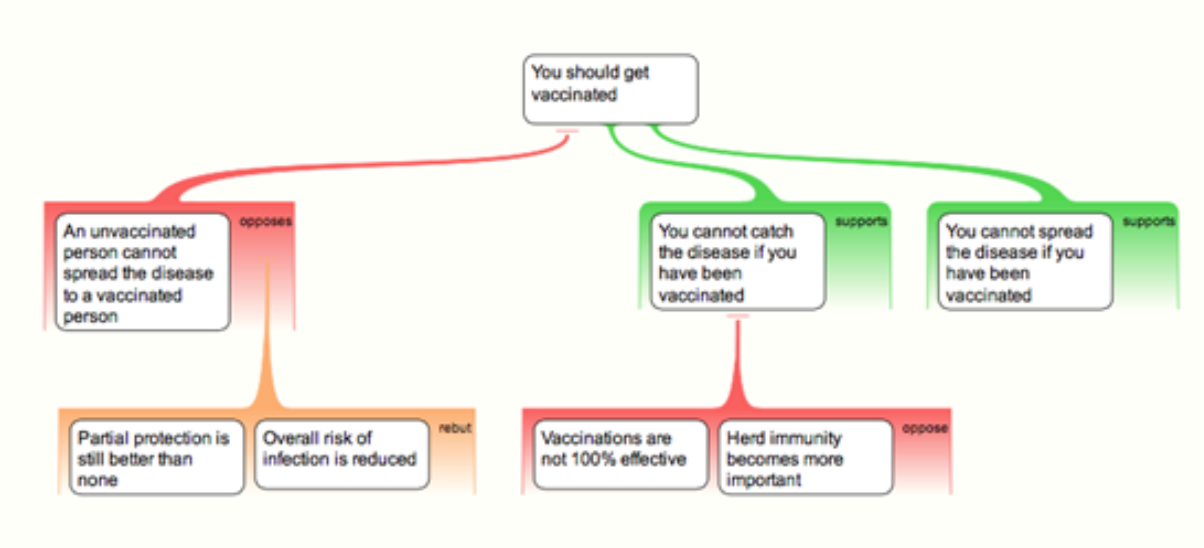

Figure 2.3: Rationale screenshot showing an argument map about vaccines

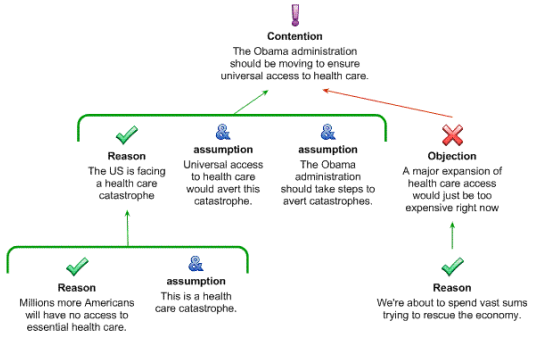

Figure 2.4: bCisive screenshot showing an argument map about Obama administration

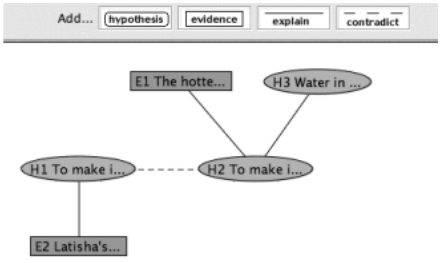

Figure 2.5: Convince me screenshot showing various visualization marks

with the purpose of increasing the students' critical thinking skills and reasoning skills.[1]

Two years later, van Gelder created bCisive[2], which is more focused in business decision making, which differentiates it from the others. This tool is, therefore, intended for a professional market and it is used to bring clarity to open questions. In bCisive, each node includes an icon on top corresponding to its type. For example, objection nodes have a cross icon while reason nodes have a check mark icon, as can be seen in Fig.2.4.

Another argument mapping tool created in the early years of this century is Convince Me, illustrated in Fig.2.5. This tool, designed to teach scientific reasoning, shows different visualization marks. Nodes are displayed as either rectangles for evidences or ellipses for hypotheses and the undirected links between them change between dashed lines for contradictions and plain lines for explanations.

[1] Rationale vs. bCisive: https://www.reasoninglab.com/product-comparison-2/
[2] bCisive website: https://www.bcisiveonline.com/

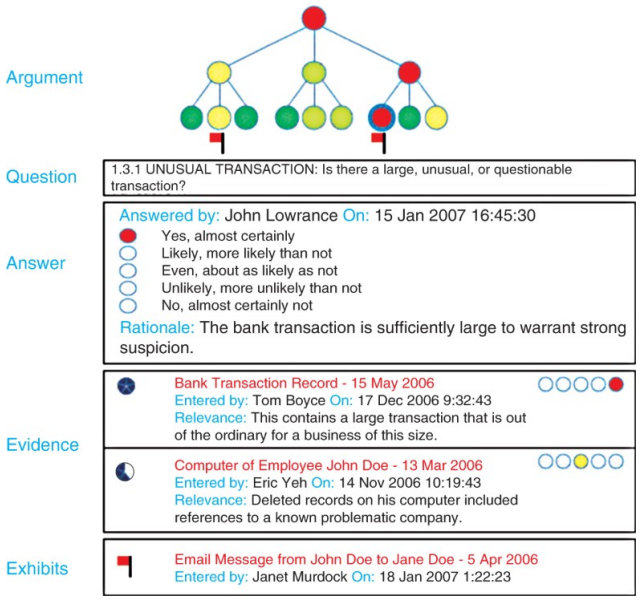

Figure 2.6: SEAS screenshot showing the supporting information for a primitive question

In 2008, the Structured Evidential Argumentation System (SEAS) [7] was developed with the purpose of assessing a specific situation using an hierarchy of questions. The argument itself is still represented by a tree, shown in Fig.2.6, but each node is a question. Leaf nodes correspond to primitive questions that are multiple choice that use a continuous scale, with one end representing strong support for a given proposition and the other representing strong refutation. The answers to these questions are translated to colors between green and red and passed to the upper nodes until a conclusion is drawn for the higher question. Each leaf node can also contain evidence and exhibits to support the given answer.

Much of the recent argument mapping tools use the same visualization techniques as the previous ones, only changing where they use colored elements and other aesthetic aspects. MindMup[3], illustrated in Fig.2.7, and OVA[4], shown in Fig.2.8, are some of these tools. OVA, however, allows the user to upload a text and creates an argument map from it that can later be edited.

A tool from 2017 called Argdown provides a Markdown inspired syntax for the user to specify and annotate arguments which it then represents graphically. As seen in Fig.2.9, it uses the same visualization techniques and aesthetic aspects as previously discussed tools.

Another recent tool, created in 2017, is Kialo[6], which is a platform designed specifically for rational debates. As shown in Fig.2.10, it mainly focuses on showing the pros and cons of each argument. For this, it presents a structure of an argument map on the top of the screen where each node is clickable. When it is clicked, it is possible to see the argument's text and all the subsequent arguments in two lists,

[3]MindMup website: https://www.mindmup.com/
[4]OVA website: http://ova.arg-tech.org/
[5]Argdown Sandbox: https://argdown.org/sandbox/html
[6]Kialo Edu website: https://www.kialo-edu.com/

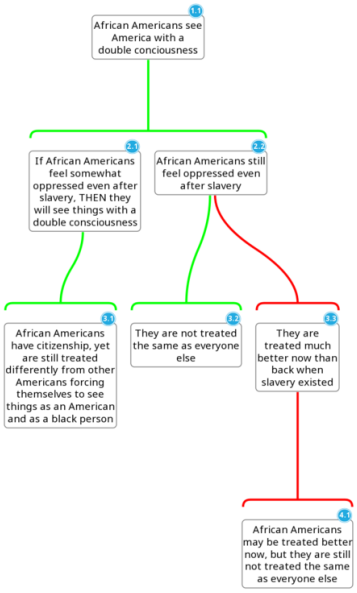

Figure 2.7: MindMup screenshot showing an example of an argument map

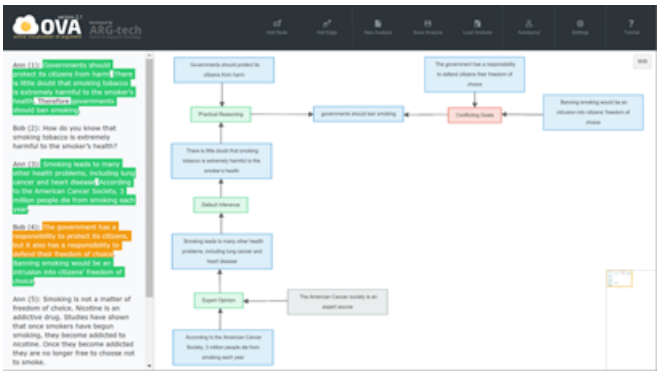

Figure 2.8: OVA platform screenshot showing an argument map based on a dialogue

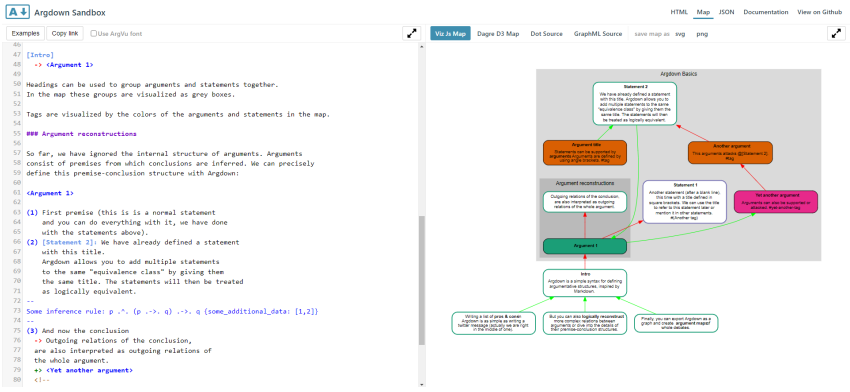

Figure 2.9: Screenshot of Argdown Sandbox[5]

Figure 2.10: Screenshot of an argument in Kialo showing its pros and cons

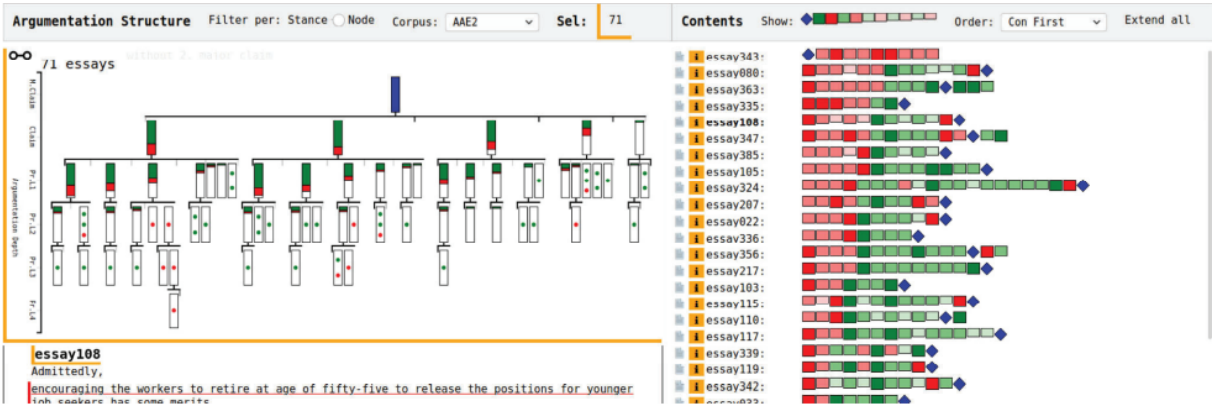

Figure 2.11: Screenshot of the visual analysis system for essays. The ArguLines are on the right, the AUOT on the top left and the text view on the bottom left

one stating the pros of the chosen argument and another stating the cons. On top of that, it also allows the users to rate the argument according to its strength, which then rearranges the corresponding list.

A 2020 argument visualization [8] uses a different approach to explore, analyse and compare argument structures. This visualization, that can be seen in Fig.2.11, consists of three components. One is called ArguLines where it is possible to see an overview of the essays. Each line represents an essay in a series of glyphs that encode its argument units, the unit's argument depth, stance and order of appearance. This component can be ordered in different ways allowing the easy and quick comparison between documents, thus providing means to reveal patterns or outliers.

The other created component is called the Argument Unit Occurrence Tree (AUOT), where subsets of essays can be analysed in detail. This component aggregates the argument structures using hierarchical histograms. For each node in the AUOT, it is possible to see the fraction of essays with a major claim in case these are blue, or the fraction of pro (green) and con (red) claims or premises (white). The AUOT facilitates the analysis of a subset of essays where it is possible to gather insights about the their commonalities, differences and trends. The last component of this visualization is the text view, where it is possible to see the full text of one or more essays. Within the text, arguments units are underlined with the corresponding color of its stance (blue, greed or red). Besides that, there are vertical bars on the left side encoding the same aspect.

Although these last two use different visualization techniques compared to the others, argument mapping tools always resort to argument maps to show the structure of an argument. This approach has been revealed as useful in the educational field as well as in the professional field. Studies, old [9] and new [10] [11], mainly using Reason!Able in an academic environment, concluded that argument mapping improved critical thinking skills and reasoning skills of the students involved. However, none of these tools gather information about the authors of the claims neither have any chronological relations between the arguments.

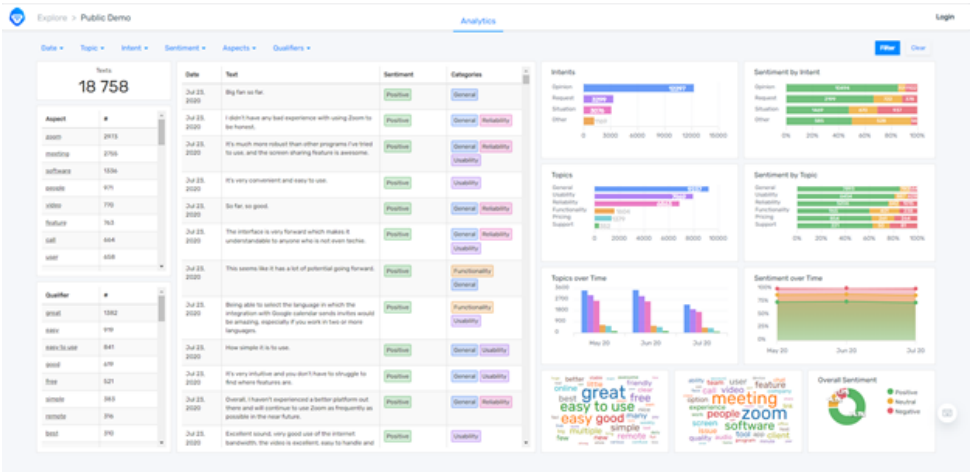

Figure 2.12: Screenshot of MonkeyLearn demo

It is possible to conclude that, relating to argument mapping, there has not been a great evolution in terms of visualization techniques. To have a better understanding of how to visualize more information about the arguments it is necessary to expand the search to non-argumentative text visualizations.

2.3 Text Visualization

Non-argumentative text visualizations offer a panoply of representations to show text characteristics that might be present in argumentative text. Tools for non-argumentative text explore different characteristics, such as stance, sentiment, topic and time in insightful ways.

One tool that allows such analysis is MonkeyLearn, a no-code interface founded in 2014 that helps create visualizations using AI technology for text analysis. It is used for unstructured data like tweets or emails and provides different visual components for using information present in that data, such as the date, the topic and the sentiment. The shown image, Fig.2.12, captures an example of a visualization that this tool can create, where the organization of the filters should be highlighted, since it gathers the categories in the same line, occupying less space, while still providing an intuitive and simple way of filtering. When a filter is selected it is shown below the category name in small letters.

Another important component used is the Sentiment by Topic graphic. For instance, this graphic can demonstrate which topics the authors are most opposed to, which is an interesting aspect to consider. However, it is not possible to convey information about the authors themselves in this example, like their age, gender or area of work. This, of course, is a setback since there is an interest in seeing who or which groups participate in which texts.

Similarly, Voyant Tools, a free, open-source application founded in 2003 that consists of 28 text anal-

Figure 2.13: Example of Collocates Graph tool

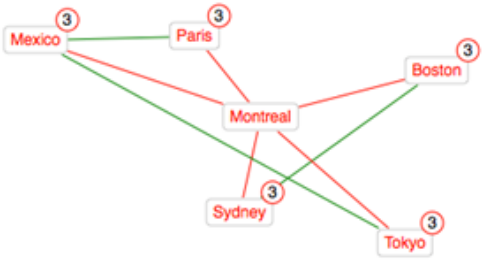

Figure 2.14: Example of RezoViz tool

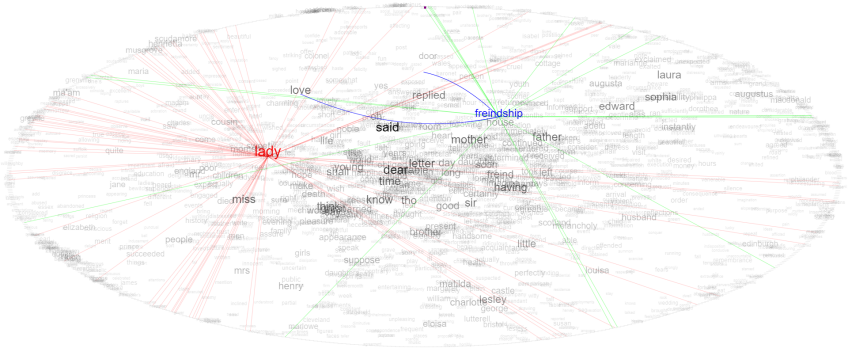

Figure 2.15: Example of TextualArc tool

ysis and visualization components[7], allows joining multiple of these components to create personalized visualizations. With them it is possible to show meaningful information about the text and the authors.

Among the components that should be highlighted, there's one called Collocates Graph, seen in Fig.2.13, which consists in a forced network graph that demonstrates how terms in the text occur in close proximity. Though the information itself is not relevant in our context, the way that it is conveyed is interesting. Another important component is RezoViz (Fig.2.14), which is a network graph that represents the connections between people, organizations, and places. TextualArc (Fig.2.15) is also an interesting component that demonstrates the distribution of terms along the document. This is represented by an ellipse that corresponds to the document and terms that are positioned according to where they appear most often. This last component, however, can produce too many terms and become unreadable.

Another insightful tool is ToPIN [12], created in 2016, that visualizes time-anchored comments in online educational videos. As seen in Fig.2.16, it focuses on the timecode of the comment, its topic, its type, and the sentiment behind it. Most of these are found by using text processing and machine learning techniques. Time is represented by a horizontal axis while topics are shown as boxes; the bigger the box the more popular the topic is. The comments are represented by lines that connect the boxes to the timeline. Comments unrelated to the video are separated from related ones having each in a different side of the timeline.

[7]List of Voyant Tools: https://voyant-tools.org/docs/#!/guide/tools

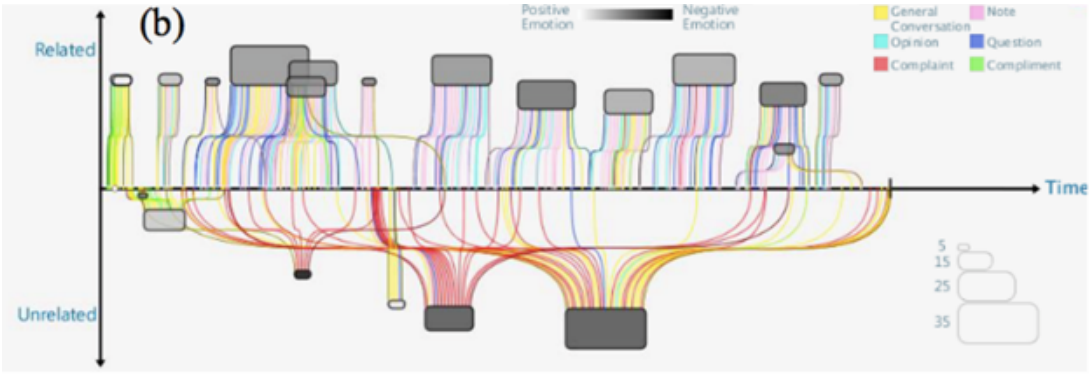

Figure 2.16: Screenshot of ToPIN showing learners' time-anchored comments on a philosophy course

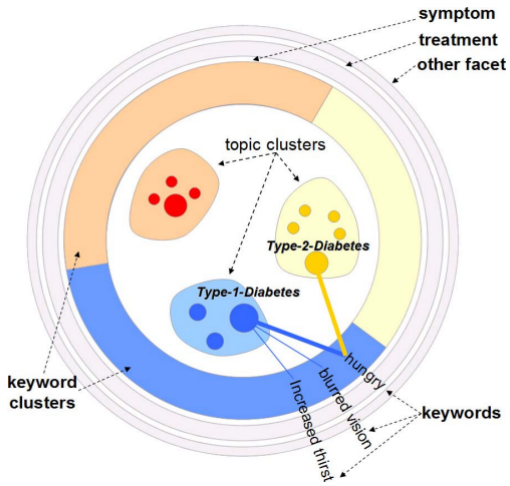

Figure 2.17: Visual encoding of SolarMap showing the different facets and keywords for the topic facet "Disease"

With this, ToPIN allows the course instructors to know the students' learning status and quality of learning. However, one is only able to convey how the students in general are learning since there is no information about a particular student. Another aspect is that it is not possible to see the comment itself, which would be important.

Another work related with topic analysis is the SolarMap [13], a multifaceted visualization created in 2011, which makes topic exploration and analysis easier. This visualization consists of topic and keyword facets and topic and keyword entities. A topic facet is always selected and it is the general concept that created the visualization. In Fig.2.17 the selected topic facet is "Disease". Topic entities in this example are the diseases themselves and they are represented as circles. If these topics appear on the same document they are clustered, using as distance their internal relations.

Keyword facets are visually encoded as surrounding grey rings. When a keyword facet is selected it opens and shows various keyword entities; in this example the keyword facet "symptom" is selected and it shows keywords like "hungry", "blurred vision" and "increased thirst". Keyword entities are organized in

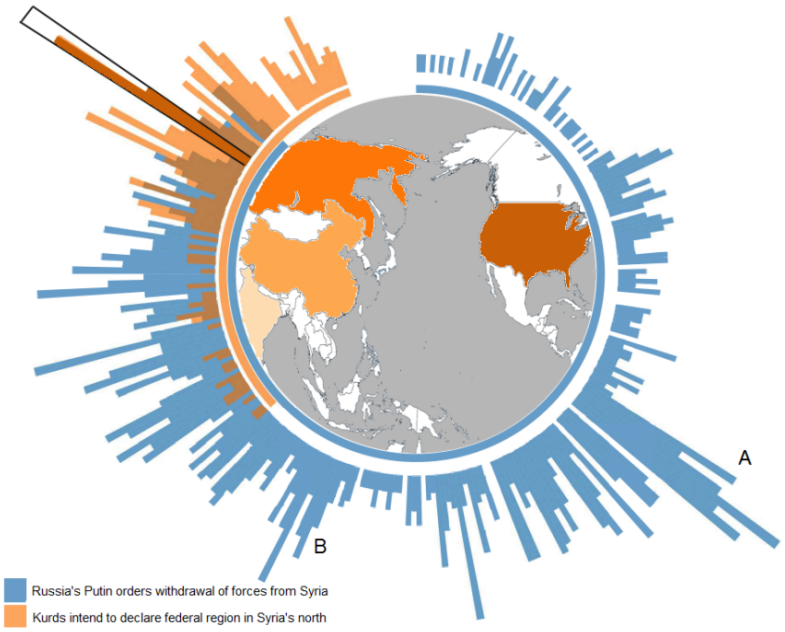

Figure 2.18: TimeRayMaps visualization showing two episodes in the news story of the Syrian war. Highlighted region shows that the episode of "Kurds declaring federal region in the North" is increasingly discussed by Russian and US sources.

color-coded clusters too, meaning they appear in the document with the same topic cluster color. Topic and keyword entities are connected by lines, where its thickness represents the number of topic entities are related to the same keyword entity.

This visualization allows the analysis of the relation of topics inside a specific document or between multiple documents. For that, it is possible to click in multiple topic entities to see the similarities and differences between the connections to the keyword entities. For example, in Fig.2.17, it is possible to see the common symptoms of two different diseases and the differences between their causes. Since SolarMap focuses on exploring the relations between topics there is no access to the analysed text, only to its keywords, which might not give enough context.

TimeRayMaps [14] is another text visualization, created in 2017, that allows the exploration of the spatial and temporal evolution of news stories. As seen in Fig.2.18, it consists in an interactive choropleth map to encode the countries of the news sources and in a radial axis representing the total observation time of the stories. Bars corresponding to the number of news articles relative to the story are positioned along the radial axis.

A news story can also have various news episodes. A news episode consists in the several reports from different sources all reporting on the same news event. These episodes are encoded by different colors with a semi transparent opacity. The number of articles related to the story and its temporal evolution can indicate the story's popularity, importance and duration while the spatial evolution may

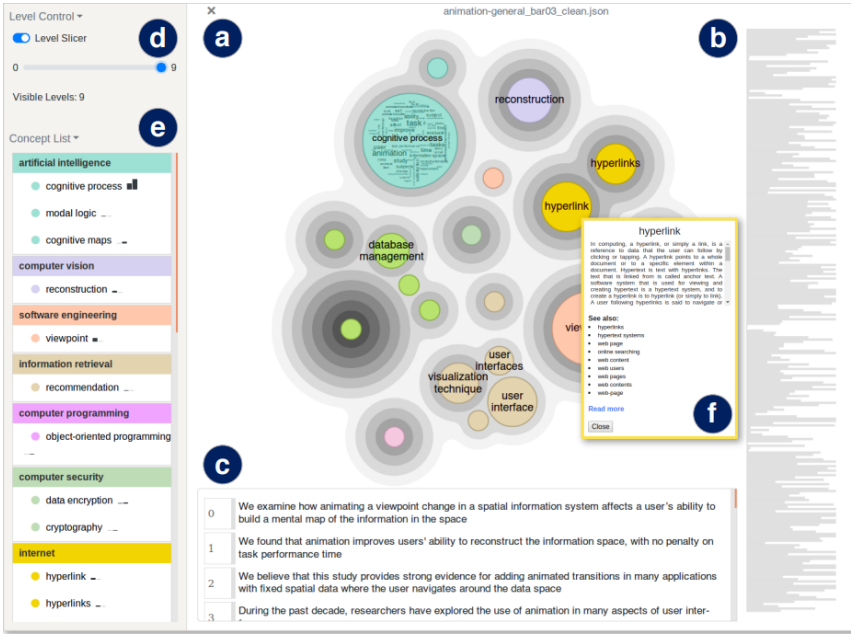

Figure 2.19: The ConceptScope interface representing a research paper discussing animation. It consists of a Bubble Treemap (a) providing a conceptual overview; the transcript (b) and text (c) views; a level slicer (d) and a list presentation (e); and a tooltip (f) with more details about the concept

indicate if a story is more local or worldwide.

However, this visualization does not focus on the evolution of sentiment and topic related to the news story. It also does not consider the text itself, only attributes like name, source and date.

Lastly, a more recent visualization, ConceptScope [15], created in 2021, allows the analysis of the conceptual structure of one or more documents. As it is possible to see in Fig.2.19, it uses a Bubble Treemap to show concept hierarchies and the relationships between concepts within the document. Each bubble represents a concept and has a word cloud inside it that represents text from the document that relates to the concept. This word cloud and the labels of each bubble only appear if there is enough space in the circle. If there is not, it is possible to zoom in. Each bubble has various outer contours that represent its parent concepts which do not explicitly appear in the document. The level of the parent concept can be chosen in a slider called Level Slicer.

This visualization also supports a transcript view, where the document is seen as a series of horizontal lines; and a text view where it is possible to see the text of the document. For each concept, it is possible to see its definition and the links to the relevant concept page on DBPedia as well as to other related concepts that may not be present in the document but can give some context to the users.

ConceptScope also allows the comparison of multiple documents side by side. With these features, this visualization helps both novices and experts in the domain, by providing an accessible conceptual overview of the document. However, ConceptScope is more suitable for analysing documents that are only related to one topic instead of those that span multiple disciplines. Since this visualization focuses

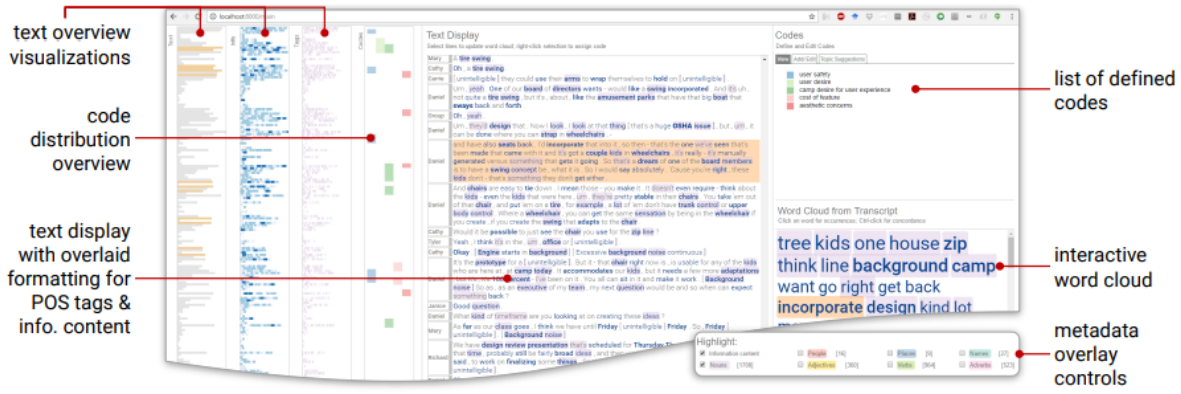

Figure 2.20: Textplorer interface showing the different text overviews on the left, the code distribution overview, the text display, the list of generated codes, the word cloud and the metadata filters

on the conceptual structure of a document, it does not analyse aspects like sentiment or the author.

Textplorer [16] is a text analysis tool created in 2017 that uses natural language processing techniques such as Parts-Of-Speech (POS) tagging, retrieving information content, and topic modeling to structure and associate different parts of data. The interface, shown in Fig.2.20, consists in various coordinated overviews that allow the identification of concepts and the relationships between them. This tool has as objective aiding Grounded Theory methodologies. Grounded Theory is an inductive method used to form theories based on the data itself and not in other pre-existing theories or concepts.

The first stage of Grounded Theory practice is called open coding where the analyst generates descriptive codes that represent different concepts. Textplorer focus mainly on this stage by allowing the creation of codes on the fly.

Besides the code distribution overview, the interface provides a transcript overview as well as a part-of-speech overview and an information content overview. It also allows the visualization of the text itself with the names of the speaker on the left of the corresponding sentence. On the right, there is a detailed word cloud that reflects the frequency of word occurrence.

Textplorer allows an analyst to identify patterns and anomalies by exploring the data using different overviews. However, and although this tool shows the speaker of each sentence, there is not information about them and it is not possible to create relationships between them.

StanceVis Prime [17] is a 2020 visualization created for the analysis of stance and sentiment in temporal text data from several social media data sources. As it is possible to see in Fig.2.21, this visualization is composed by several components.

The first component (a), is the loaded data series table, that includes their sparkline previews. The next component (b) consists of data series similarity views. On the left, a temporal slider can be found as well as a 2D projection view representing the data series at the current time step. On the right, a temporal similarity view showing the data series' projections over time can be found.

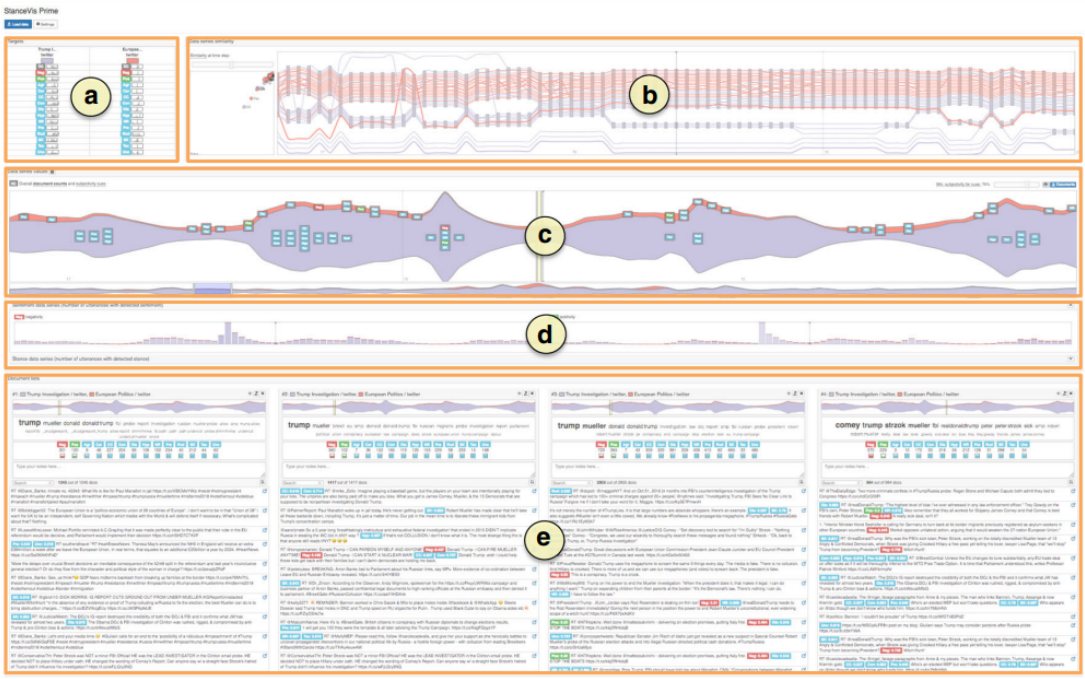

Figure 2.21: StanceVis Prime visualization showing its multiple views

Below these two components, there is a stacked graph indicating how many documents are related to a specific target/domain combination, for example, Brexit/Reddit. On the lowest part of the component, an overview for the complete loaded dataset can be found, which also serves as a range slider. The selected area appears on the top part of the component. This zoomed in area shows subjectivity labels indicating sentiment (in blue) or stances (in red, in case of negativity or green, in case of positivity)

This visualization also allows to see how many declarations were made with a specific sentiment or stance. To achieve this, a series of stacked bar charts (d) were created for each sentiment and stance.

Finally, the document list views can be found at the bottom of the visualization. Each view includes a stacked graph similar to the previous one, a list of important concepts found in the document's text, statistics for sentiment and stance classification results, an area for user notes and a document view with labels indicating the sentiment and stance classification results of the declaration in question.

StanceVis Prime allows the analysis of temporal trends in regard to stance and sentiment data series. It is also possible to investigate similarities of multiple data series over time. However, there is no information about the authors of the documents and how that can influence sentiment and stance.

2.3.1 Storyline visualizations

Other works relate to story visualizations. StoryFlow [18] is a storyline visualization, which illustrates the dynamic relationships between entities in a story. Each entity, illustrated in Fig.2.22 is represented by a line going from left to right along the horizontal timeline; when multiple lines are bundled together

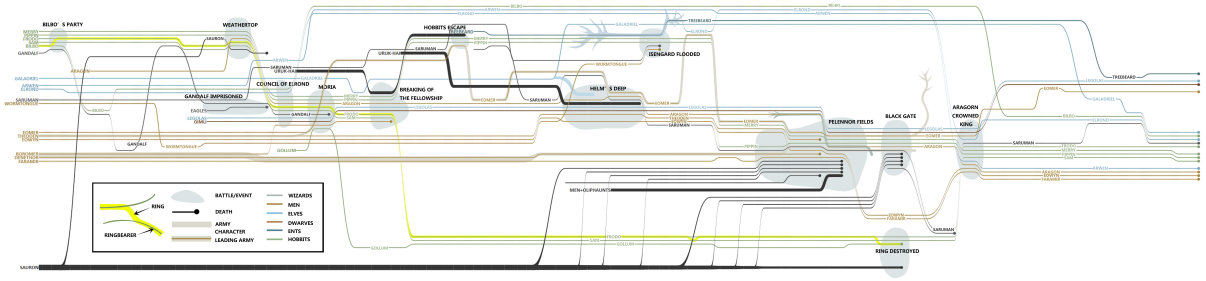

Figure 2.22: Storyline visualization created by StoryFlow about the movie The Lord of the Rings

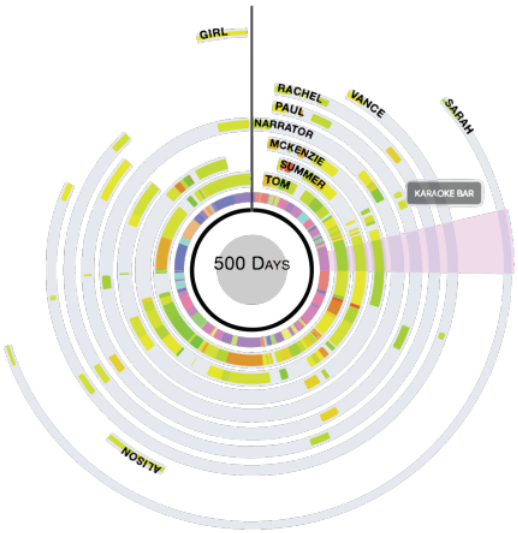

Figure 2.23: StoryPrint visualization for the *500 days of Summer* movie showing the emotions of each character

during a time period it means the entities are interacting with each other; if the lines converge or diverge it indicates the interaction is starting or terminating, respectively.

This visualization stands out from other storyline visualizations because of its efficiency and interactivity. For instance, it can construct an aesthetically-appealing storyline in less than one second, even with hundreds of entities and hundreds of time frames. It also provides various interactions such as aggregating lines and straightening a line, allowing for real-time exploration of the relationships within a big and complex dataset.

While other storyline visualizations can get a bit confusing with too many lines, StoryFlow implements a layout optimization that joins discrete optimization to minimize the number of line crossings and continuous optimization to minimize the white space. Although it is possible to see the evolution of topics along the timeline, the text corresponding to what the entity said in that moment is not accessible.

StoryPrint [19] is an interactive visualization similar to StoryFlow. As illustrated in Fig.2.23, it shows a story and the participation of the characters as well as setting changes and character's sentiment throughout the story. StoryPrint goes beyond StoryFlow with a completely different encoding for script-based media. The visualization consists in an inner ring and various outer rings called character arcs.

The inner ring is partitioned into segments, which are ordered chronologically. Each segment represents a scene in the story and its relative length along the ring's circumference. For each segment there is a color associated with the setting of the scene.

The character arcs represent the different characters of the story. Each arc indicates the scenes a character has talked in as well as maps the estimated emotional experience for a character to a hue between red (negative experience) and green (positive experience). This estimation is done using sentiment analysis for the character's lines on a scene-by-scene basis.

It is also possible to view a summary of the emotion of a specific character throughout the film by clicking on the character's arc.

These features allow the analysis of a script without having to read it multiple times, thus saving time. This visualization also provides an overlay feature which allows the comparison of two stories, showing the characters that have been removed or added in each scene. This comparison is beneficial, for example, when analysing two versions of the same movie or two episodes of the same TV series.

StoryPrint allows the visualization of characters' presence and sentiment as well as setting changes while also providing means of comparison between two stories. However, it is not feasible to compare more than two stories using the created overlay neither it is possible to use this tool with stories that do not follow a specific script structure. Moreover, it does not provide access to the original script to know what the characters said in the selected scene.

2.4 Storytelling visualizations

These type of visualizations are a creative way of telling a story based on visual components. Most of them use scrolling as a trigger event to change the state of the visualization. Others only show a visual component with no interaction and take conclusions from its current state. However, all of them consist in visual elements joined with text that explains them as well as what should be taken out of the visualization. First, it is important to know that this type of visualization is placed along a spectrum of author-driven and reader-driven approaches. In the following section, these approaches and the visualizations schemas resulting from them will be discussed.

2.4.1 Storytelling approaches

Storytelling visualizations must balance between a author-driven and a reader-driven story. Author-driven means the story has a specific and linear order of events and focuses more in the message that is conveying rather than the interactivity that it can provide. Contrary to this, reader-driven stories promote a more exploratory way for the reader to comprehend the data, without focusing so much in a specific story itself.

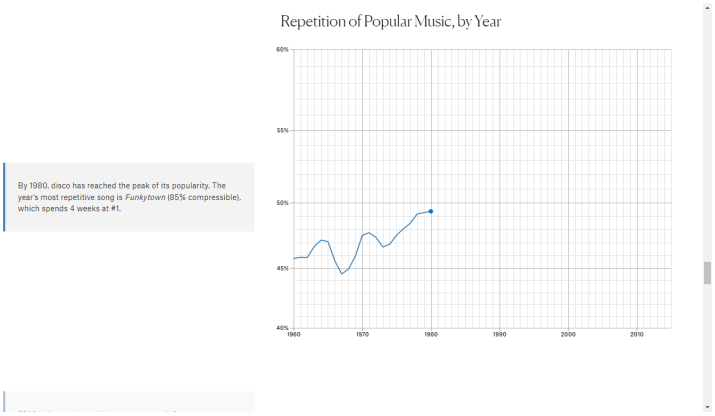

Figure 2.24: Screenshot of one of the visual components of the ”Are Pop Lyrics Getting More Repetitive?” page

Using these approaches together, there are multiple possible combinations for storytelling visualizations. However, some of them became the most common. Jeffrey Heer along with Edward Segel [20], present three of the most used schemas.

The first model, Martini Glass, prioritizes the author-driven approach. The name of the schema is due to the fact that, initially, the visualization is guided by questions and observations created by the author. Afterwards, in the widening mouth of the glass, the reader can interactively explore it.

The second model, Interactive Slideshow, gathers both approaches, by following a slideshow structure, and, contrary to the previous model, it provides some interaction mid-narrative, which allows the user to explore the data before moving on to the rest of the story.

The final schema, Drill-Down Story, focuses more in a reader-driven approach, by presenting a general theme first and allowing the reader to choose between instances of that theme to learn more about it. This provides a way for the user to read the stories they want, when they want.

Next, some examples of storytelling visualizations will be discussed to learn the differences between them and how they fit in these models.

2.4.2 Examples of storytelling visualizations

The Pudding[8] gathers several visualizations of this kind, turning it into a big and broad library of different storytelling visualizations. One of the pages is named ”Are Pop Lyrics Getting More Repetitive?”[9] that discusses if pop music has become more repetitive throughout the years. The page is composed of multiple visual components. Some change while scrolling through the page, for example, showing links between elements through animations or showing more data in a chart, as shown in Fig.2.24. These changes are accompanied by a text that is next to the component. The changing components, however,

[8]The Pudding: https://pudding.cool
[9]Are Pop Lyrics Getting More Repetitive?: https://pudding.cool/2017/05/song-repetition/

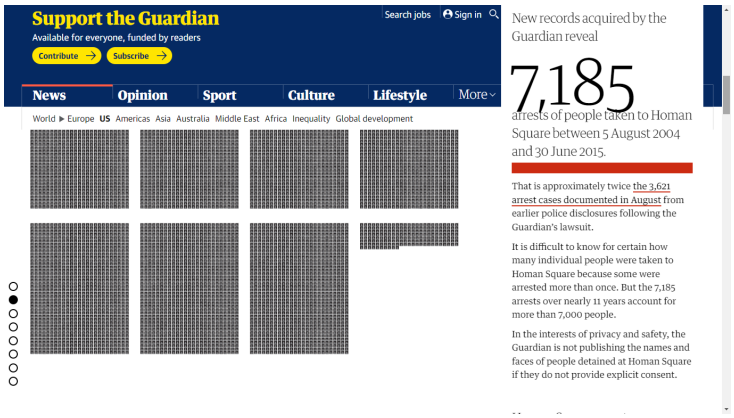

Figure 2.25: Screenshot of The Guardian's page, "Homan Square: A portrait of Chicago's detainees"

can not be interacted using buttons or even hovering. Nevertheless, this page has components that are interactive through buttons but do not have any animations and do not change when scrolling.

Another page of this website called "Can Data Die?"[10] follows the same pattern. It first shows sequence of images and scrolling text to tell how images circulate through the internet. Afterwards, it shows a bar chart about the Lenna image and its appearance throughout the years, that changes through scrolling till the end of the page, where it can be hovered to get more information.

The Pudding also has a page named "Twenty Years Of The NBA Redrafted"[11]. While telling a story about NBA, it gathers multiple animations that make it more interactive. Either the components change through scrolling, to tell a specific story, but, contrary to the previous page, allowing hovering, or they are static, meaning they always show the same data, but are also interactive through hovering for a more exploratory analysis.

The Guardian[12], although it is known as a newspaper, also created pages with storytelling visualizations. One of these pages is called "Homan Square: A portrait of Chicago's detainees"[13], that tells the story about the Chicago's police detainees. As seen in Fig.2.25, this page is composed of a single visual component composed of multiple squares representing the detainees that change position when scrolling the text element that appears on the right side of the page.

Another example of a storytelling visualization of The Guardian is "How China's economic slowdown could weigh on the rest of the world"[14], where, contrary to the pages seen till now, the scroll does not have an effect in the visual components. Nevertheless, some parts of the text work as a button to change the visualization input. Overall, it is still an example of this kind of visualizations since it gathers visual components and text to explain a big story.

[10] Can Data Die?: https://pudding.cool/2021/10/lenna/
[11] Twenty Years Of The NBA Redrafted: https://pudding.cool/2017/03/redraft/
[12] The Guardian: https://www.theguardian.com/us-news
[13] Homan Square: A portrait of Chicago's detainees: https://www.theguardian.com/us-news/ng-interactive/2015/oct/19/homan-square-chicago-police-detainees
[14] How China's economic slowdown could weigh on the rest of the world: https://www.theguardian.com/world/ng-interactive/2015/aug/26/china-economic-slowdown-world-imports

Other examples of pages with scrolling as trigger event, also called scrollytelling visualizations, can be found. For instance, the R2D3 website includes a visualization called ”A visual introduction to machine learning”[15] that has multiple animations when swapping from one visual component to other. Similarly to other pages, the text is next to the components and its change is what activates the new elements in the visualization.

Another example is Muyueh’s website that has ”Green Honey”[16], a very colorful storytelling visualization that explains how colors are differently perceived by English or Chinese cultures. This page works as the first page of The Guardian, where there is a single visual component that changes state using animations when scrolling the text component.

Bloomberg also has a page called ”Bubble to Bust to Recovery”[17] with an interactive line chart about US housing marketing. This visualization is not exactly triggered by scrolling but rather by clicking on left or right buttons. Besides that, it is possible to use hovering to know the exact numbers of the points in the line chart. The component then changes according to what they want to show and a text on top of it appears explaining the conclusions to take from the chart.

A different storytelling visualization is called ”Organisation for Economic Co-operation and Development (OECD) Better Life Index”[18]; this index allows the comparison of well-being across countries, based on topics the OECD has identified as essential, in terms of material living conditions and quality of life. In this visualization there are multiple visual components separated in tabs accompanied by text below them. The elements that compose them can be clicked in order to get more information about them. After clicking them, the visual component changes state as well as the text, which explains the findings of the visualization according to the clicked element. This way, the state can be changed in various manners by clicking in different elements.

There are many more visualizations like these, that vary on how the visual components react to interaction, which can include scrolling, clicking, hovering, among others.

2.5 Discussion

In this section, previously explored visualizations will be critically discussed, in order to understand the importance they have when creating the proposed solution. Table 2.1 shows the different analysis argument and text visualizations can offer.

The first ten visualizations are focused on argument analysis, which, in most cases, means that it provides an argument map for the given text. As seen before, argument maps are the standard way of visualizing the argument logic structure, in some cases, showing the attack and support connections.

[15] A visual introduction to machine learning: http://www.r2d3.us/visual-intro-to-machine-learning-part-1/
[16] Green Honey: http://muyueh.com/greenhoney/
[17] Bubble to Bust to Recovery: https://www.bloomberg.com/graphics/dataview/bubble-to-bust-to-recovery/
[18] OECD Better Life Index: https://www.oecdbetterlifeindex.org/

Visualization Name	Argument Analysis	Time Analysis	Stance Analysis	Author Information	Keyword/ Topic Analysis
Reason!Able	Yes	No	Yes	No	No
Rationale	Yes	No	Yes	No	No
bCisive	Yes	No	Yes	No	No
Convince Me	Yes	No	No	No	No
SEAS	Yes	No	Yes	No	No
MindMup	Yes	No	Yes	No	No
OVA	Yes	No	Yes	No	No
Argdown	Yes	No	No	No	No
Kialo	Yes	No	Yes	No	No
Visual Analysis System for essays	Yes	No	Yes	No	No
MonkeyLearn	No	Yes	Yes	No	Yes
Voyant Tools	No	Yes	No	No	Yes
ToPIN	No	Yes	No	No	No
StoryFlow	No	Yes	No	Yes	No
StoryPrint	No	Yes	No	Yes	No
SolarMap	No	No	No	No	Yes
TimeRayMaps	No	Yes	No	No	No
ConceptScope	No	No	No	No	Yes
Textplorer	No	Yes	No	No	Yes
StanceVisPrime	No	Yes	Yes	No	No

Table 2.1: Analysis types for each previously seen argument and text visualizations

This approach has been used for centuries, without evolving much. However, some visualizations extended this idea and formed new ways of showing the data, but they all resort to some kind of node-link graph to represent the links between arguments. It has been studied how argument maps can improve critical thinking and reasoning skills, specially in an academic environment. Seen that this visualization method has persisted throughout the years and it is perceived as the best way of learning about an argument, it is important that it is part of the developed platform.

However, these types of visualization do not analyse aspects that can be relevant in understanding any argument; characteristics such as time, author and topic. These are found in both argumentative and non-argumentative text, however, only visualizations for the latter involve these types of analysis. Among text visualizations, some are more focused in the analysis of keywords in order to understand the topics evolution; others focus their attention in stance and sentiment analysis. These two analysis are done using different approaches, and sometimes they are joined together in a single visual component. This approach can be used in our future visualization as it compacts two analysis in one. However, more visual components regarding these analysis separately will be needed.

As seen in Table 2.1, only story visualizations gather information about the author, and even so it is just the characters' names. It is hard to find a visualization that uses personal information of the author to take conclusions about the text.

Time analysis can have two meanings: either time passed throughout a document or time passed

throughout various documents. If only one document is analysed then the perceived time will be how the text evolves throughout the document. On the other hand, if the visualization allows more than one document to be analysed and the documents have publish dates, then there is a chronological order for them. Argument visualizations do not provide a way of knowing which argument came first in the text. Text visualizations, however, can achieve this using timelines, which can be circular or straight. This type of analysis is very important to understand the evolution of the arguments. Because of this, the proposed solution should include a timeline of some sort that gives chronological information about the documents.

Page	**Model**	**Single/ Multiple Components**	**Component changes through scrolling of text**	**Component changes through clicking**	**Static component**	**Uses animations to change state of visualization**	**Click/ Hovering Interactions**
"Are Pop Lyrics Getting More Repetitive?", The Pudding	Martini Glass	Multiple	Yes	Yes	Yes	Yes	No
"Can Data Die?", The Pudding	Martini Glass	Multiple	Yes	No	Yes	Yes	Yes
"Twenty Years Of The NBA Redrafted", The Pudding	Interactive Slideshow	Multiple	Yes	No	Yes	Yes	Yes
"Homan Square: A portrait of Chicago's detainees", The Guardian	Interactive Slideshow	Single	Yes	No	No	Yes	No
"How China's economic slowdown could weigh on the rest of the world", The Guardian	Interactive Slideshow	Multiple	No	No	Yes	No	Yes
"A visual introduction to machine learning", R2D3	Interactive Slideshow	Multiple	No	No	Yes	Yes	No
"Green Honey", Muyueh	Interactive Slideshow	Single	Yes	No	No	Yes	No
"Bubble to Bust to Recovery", Bloomberg	Interactive Slideshow	Single	No	Yes	No	Yes	Yes
"OECD Better Life Index"	Drill-Down Story	Single	No	Yes	No	Yes	Yes

Table 2.2: Behaviour types for each previously seen storytelling visualization

A different type of visualization, storytelling, was also explored in the previous section so it is possible to accomplish a more creative visualization. Multiple pages were seen, enough to understand that the components that make these types of visualizations are visual components accompanied with text that explains the conclusions that can be taken from the shown data. In Table 2.2, there can be found a list of these visualizations and their characteristics as well as their types. Some of the storytelling visualizations have a single visual component that changes or multiple components with different interactions. Overall, these visualizations can have scroll as a trigger, meaning the visual components change as the user scrolls through the text element, and in this case, they are called scrollytelling. There is a variant to this, where the component changes by clicking in buttons. To achieve a better user experience, these

visualizations use transition animations when changing its state.

Overall, the included visual components may or may not be interactive, providing or not clicking and hovering interactions to provide a better way of exploring the data.

In conclusion, multiple visualizations of different types were explored, in order to make a creative and intuitive solution. Argument maps remain an important visual component to have and time, stance and topic analysis should also be considered relevant for the future platform. Storytelling visualizations can provide a better and more engaging way of telling a big story taken from argumentative text, but the way it works depends of the conclusions that can be reached.

3

Solution

Contents

3.1 Framework's Previous Version 29
3.2 Requirements 32
3.3 Architecture and Technology 33
3.4 Visual Argument Framework 34
3.5 Dashboard 43
3.6 ScrollyTeller 50

As observed in the previous section, which presents the state of the art, the visualization tools regarding the analysis of argumentative text have not evolved significantly over the years. Argument maps are still the most used visual representation for arguments. However, text visualizations reveal solutions far more interesting, which allow the analysis of other characteristics, that can also be present in argumentative text. Therefore, there is a great potential in using these ideas in a platform to analyse these type of text. In fact, there is not yet a platform that uses visual components of this type to facilitate the analysis of argumentative text.

In this chapter, the solution will be presented, which was created on top of the VA Framework, a framework that was developed by a colleague, João Oliveira, as part of his thesis work [2], which allows the creation of multiple visual components. The previous version of this framework will be shown, where it will be explained how the VA Framework works and the technology it used as well as it will be discussed the limitations that lead to the framework's improvement.

Those improvements will be presented afterwards, which consist in extending and adapting the VA Framework to real datasets. To show the versatility of this new version, two demonstrators were created for the different case studies, using combined approaches from the related work.

3.1 Framework's Previous Version

As previously noted, the solution was developed on top of an improved version of the VA Framework, since the previous one had some constraints. In this section, its architecture as well as its limitations will be discussed.

3.1.1 Architecture and Technology

The VA Framework followed a client-server model. The server consisted of an Application Programming Interface (API) and a database, which gathered multiple JavaScript Object Notation (JSON) files, each representing a parliamentary debate. The API provided two Hypertext Transfer Protocol (HTTP) requests, GET and POST, to interact with the database and pass the data between both.

The client side consists of, besides the VA Framework itself, a JavaScript module responsible for asking the necessary data to the API and passing it to the framework.

To create the visual elements, the framework uses Vue.js[1] as well as D3.js[2]. D3.js is a popular and reliable library that allows the creation of the visual components, making use of HyperText Markup Language (HTML), Scalable Vector Graphics (SVGs) and Cascading Style Sheets (CSS). In a complementary manner, Vue.js provides means to construct the framework structure with a good and scalable

[1] https://vuejs.org/
[2] https://d3js.org/

```
<map-component :documents="documents" id="map-component" :config="{file: 'map_config.js'}"></map-component>
```

Figure 3.1: Example of an HTML element corresponding to a visual map component of the framework

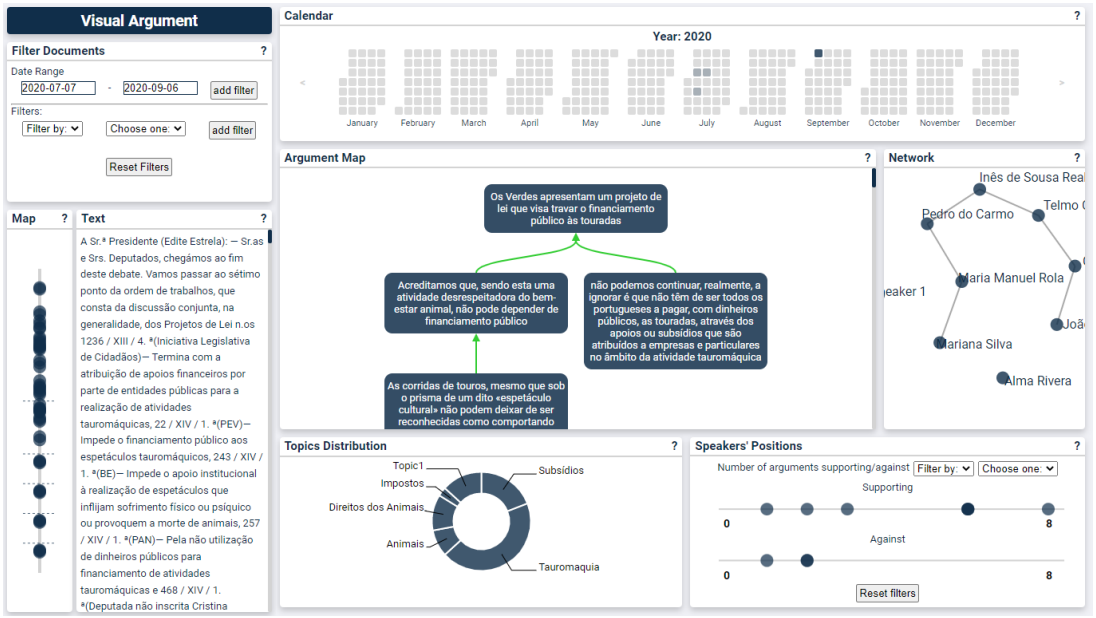

Figure 3.2: Screenshot of the previous version of the Visual Argument Framework

design, which can easily be integrated with the D3.js library.

The framework's components can pass data between each other, allowing interaction, due to a Vue instance, called Event Bus. To pass the various JSON files, a dummy document was created, result of the merging of all the documents, chronologically ordered.

Each component of the framework is also a Vue instance, which contains its data, its methods and its lifecycle hooks. To take advantage of the Vue.js framework, each component was developed as modular and independent so it could be integrated in any part of any HTML page. To add a component, the developer must insert an HTML element with a tag corresponding to the name of the visual component. This line of code, similar to the one shown in Fig.3.1, should also include attributes, such as the id and the name of the configuration file and the array of documents. The configuration file allows to define parameters for that specific component as well as configure how it interacts with the others.

The VA Framework allows multiple layouts, since every element is independent. One of the possible layouts, including all the developed visual components, can be seen in Fig.3.2. As it can be observed, it is possible to access the raw text of the documents as well as their argument maps, which enables the learning of the structure of an argument.

Other visual components provide different information that can help reach conclusions about the data itself. For instance, on the right, it is possible to see the relations between speakers and how many arguments they have had in favor or against another argument. On the bottom-middle, information about the topics is shown, to understand their distribution throughout the documents.

Chronological data can also be found. The platform provides a timeline for the arguments along the documents, on the left, and a calendar heatmap for documents, on the top, which, joined with other visual components, can show the distribution of a topic or a speaker over time. However, this version of the VA Framework presents various limitations, specially regarding the data it is conveying, which will be explained next.

3.1.2 Limitations

As said before, this version of the VA Framework received mock-up data from parliamentary debates. This means the framework was not receiving real data, thus the components were not adapted to the complexity and characteristics of the actual datasets.

This was the main issue of this version, that triggered numerous changes afterwards. The real datasets consisted of large files with hundreds of debates or articles, depending on the domain. The structure of the datasets, which will be shown in the next chapter, were different than the framework expected. To accommodate the attributes and the structure of the real datasets, the visual components would have to be altered.

Besides that, this version was developed having in mind only one case study. Although the framework is modular and flexible, a lot of the components are constructed just considering the structure of one specific dataset. To use other datasets, with other structures, the framework would not be ready. Therefore, depending on the requirements of the different case studies, the framework would have to be adapted in order to be able to create components for both domains, without conflicts.

Regarding the visual components themselves, they leave some important questions left unanswered, and crucial tasks that are not easy to perform. For example, this version does not provide an easy way to know when the speakers were in favor or against an argument along the document. It is possible to see, for each speaker, the amount of supporting or against arguments, but there is not any chronological data behind it.

The whole speakers' position component can be quite difficult to understand. To see if a certain topic had more supporting or against arguments the user has to count them for each speaker and sum them. There is another task that should be simplified in the framework: seeing the evolution of topics over a period of time. This task requires selecting a topic and then having to see the stance of each argument on the timeline map component.

Overall, the documents are not treated as one big story. Although their texts are joined together, there is no way to see the relation between the documents in regards of topic, for example. This makes it feel like the different documents do not have a connection between them, which is not true.

Because of these limitations, this framework needed to be improved, so it could respond to the new requirements, obtained with the real datasets, and answer to the needed questions in a more intuitive

and easy way.

3.2 Requirements

Using as basis the already developed visualization techniques seen in the previous section, it is possible to determine the most fundamental features that the solution should have. Since each case study presents different data and has a different visualization goal, the requirements for each demonstrator will be different. However, both demonstrators must fulfill the following requirements:

1. The visual components must interact with each other, whilst being independent;
2. Both demonstrators must be usable by experts and non-experts in linguistics;
3. The platform must be responsive for computer.

3.2.1 Opinion Articles

The first case study, refers to a deeper argumentative text analysis, which means that showing information regarding the structure of the argument will be more important. The following list presents the tasks this specific demonstrator needs to allow:

1. Analyse the structure of arguments present in the document, allowing the identification of their components: ADUs and relations;
2. Identify the types and how many ADUs are present in the document;
3. Observe the distribution of ADUs in the text of the document;
4. Understand the distribution and evolution of support and attack relations throughout the document;
5. Analyse the distribution of support and attack relations per topic;
6. Perceive documents' similarity regarding their content and identify the most similar documents;
7. Compare documents, by filtering and selecting.

3.2.2 Parliamentary Debates

In this case study, the goal is to tell a specific story using the provided data. As a ScrollyTelling visualization, this demonstrator needs to follow the requirements below:

1. Convey the story in a clear and simple way;
2. Have scroll interaction throughout the visualization, which allows to go back and forth in the story;
3. Have interactive visual components with zoom, pan, click and hover events;
4. Have an exploratory area for curious users who want to know more about the theme that was not conveyed in the story.

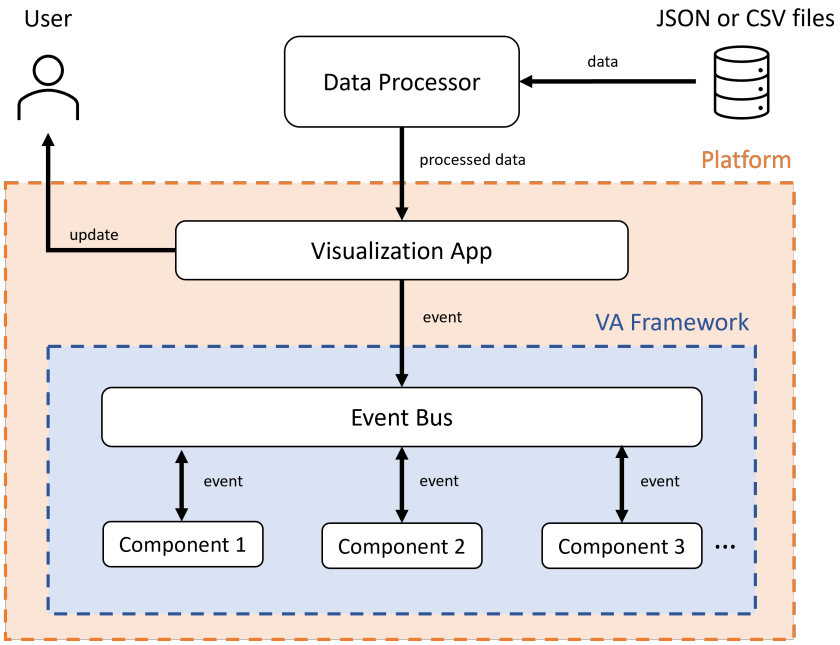

Figure 3.3: Architecture Diagram

3.3 Architecture and Technology

The developed demonstrators follow the architecture presented in Fig.3.3, where the VA Framework is shown as part of the web platform. The datasets, that will be discussed in the next subsection, are retrieved from JSON or Comma-Separated Values (CSV) files and passed to the Data Processor, which is responsible for reading and processing the data, depending on the needs of the platform.

The processed data is then passed to the Visualization App, which is specific to the platform, and it is in charge of creating the structure of the platform in question. After the platform structure is constructed, the Visualization App emits an event through the Event Bus, which, in turn, emits events to create the necessary components that exist in the VA Framework. The Event Bus also permits the transmission of data between components which allows for great interactivity.

3.3.1 Datasets

The DARGMINTS project uses as case studies three different domains: opinion articles, parliamentary debates and online political forums. However, there were only extracted and organized datasets for the first two areas.

For the opinion articles, the dataset is a JSON file, that contains multiple articles with important fields, such as, unique identifier, title, authors, text and date. There are other fields that were also explored for the visualization, such as the keywords, to see the similarity of the articles and the topics, to filter the documents and perform topic analysis.

In this dataset, it is also possible to analyse the arguments annotations for each article, where the statements and relations between them are shown. These arguments were annotated by people, called annotators, and for each article there were three annotators assigned. In this dataset, the statements are called nodes, and the relations edges.

For the parliamentary debates, there are various datasets, which are all CSV files. The datasets provided by the DARGMINTS project were pre-processed in order to create new CSV files that corresponded to the needs of the visualization for this domain.

One of these files lists all the interventions made by a congressperson in all debates. Therefore, each line represents an intervention. This dataset contains data from the initiative that was being discussed in that debate, such as, the unique identifier, the author, the title and information regarding the votes (results, favour, against and abstention votes); as well as information about the intervention, such as, the congressperson unique identifier, their party and the intervention text. There is also the date and the legislature of the debate.

The other files have similar structures. One instead of listing all interventions, only lists the initiatives, which leads to less repeated data. The structure of the datasets of both domain are presented in Appendix A.

3.4 Visual Argument Framework

As seen in the first section of this chapter, the VA Framework has its limitations. Therefore, the existing components had to be changed and new components had to be created to visualize the new attributes of the datasets. Some of the existing components were not ready for such complex and large datasets, so their structure needed to be changed almost entirely. Some components that were already created did not make sense for the provided data and were not included in the platforms.

3.4.1 Visual Components

In this section, it is presented a list of components that are shown in the developed platforms, using the VA Framework, explaining which components are new and the changes performed in case they already existed in the previous version of the framework.

3.4.1.A List Component

Since the provided dataset presents more than 300 documents, a way to navigate through these articles should be available to the user to easily select and discard a document. This component was missing in the previous framework version.

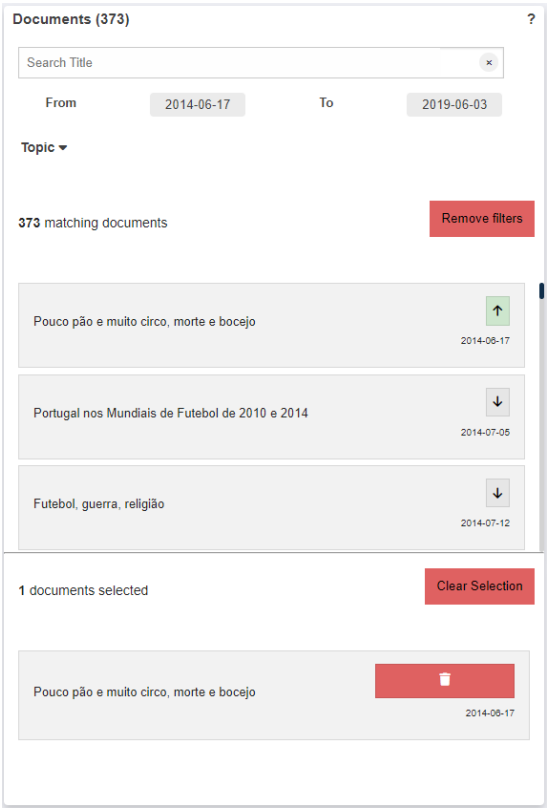

Figure 3.4: Example of the list component

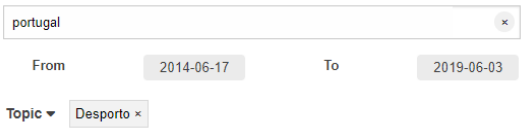

Figure 3.5: Example of the filter bar component

As seen in Fig.3.4, the list component is divided in three parts: the filters' bar, the filtered documents and the selected documents. The filters' bar will be explain in the next section, since it includes a component of itself.

The filtered documents list presents all the documents that match the filters placed in the top bar. Each rectangle represents an article and shows its title and its date. The documents can be selected by clicking on the arrow on the right-side of the corresponding rectangle. Once it is selected, the arrow changes direction and color.

The selected documents list presents all the documents that were selected. It is possible to delete the selection of a specific article by clicking on the trash can on the right-side of the corresponding rectangle. To discard all the selected documents it is possible to click on the "Clear Selection" option. To know how many documents are filtered or selected there are written indicators on top of both lists specifying it.

3.4.1.B Filter Component

To easily search the various documents, there are a series of filters that can be used to filter the list of articles, as seen in Fig.3.5. The first part of this component is the search bar used to search the title of the article. This bar cannot be used to search topics or keywords, since it only matches titles.

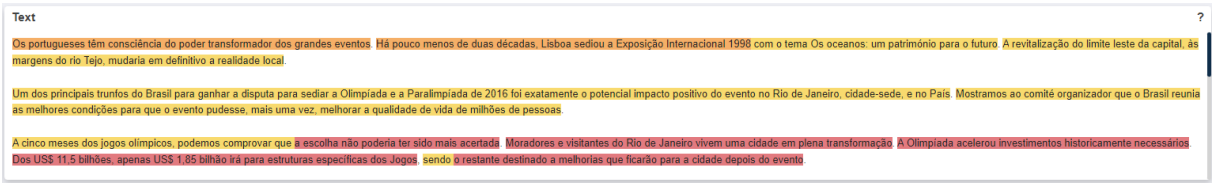

Figure 3.6: Example of the text component with all annotators selected

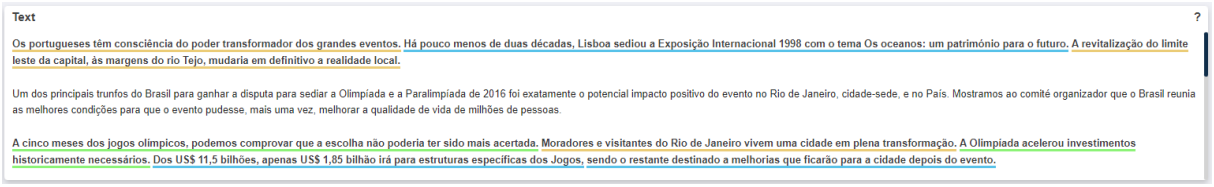

Figure 3.7: Example of the text component with one annotator selected

The second part relates to the dates of the articles, where it is possible to filter the list by date. In other words, it only presents documents that were published between the selected dates.

Finally, there is a drop-down menu to select a topic. After a topic is selected it will appear next to the menu, so the user can easily understand it was picked and can remove the selection if wanted. More than one topic can be picked to see all the documents of those specific topics. If no topic is selected, then all documents will be presented.

To remove all filters and see all the documents again, it is possible to select the button ”Remove Filters”, seen in Fig.3.4. The filter component also interferes with the Network Graph Component, explained in one of the following sections.

3.4.1.C Text Component

Knowing the text of the selected documents is a crucial aspect of any tool of this type, as seen in the previous chapter regarding the related work. Consequently, a component that shows the raw text of the document must be present in the framework.

This component was already developed but it was adapted to accommodate new aspects of the opinion articles dataset, such as the annotators. Since each article has three annotators assigned to it and each created its list of annotations with argument maps, it was to be expected that some parts of the text had more common annotations than others. These pieces of text were said to be more argumentative. To indicate this, it was created a form of heatmap, where the most intense color shows the most argumentative parts of the text.

In Fig.3.6, it is possible to see three paragraphs, where the last one has the most intense color, showing that it has more arguments in common, thus, being the most argumentative. The heatmap only appears when all the annotators are selected. If only one is selected, the component changes view, showing the annotations of the selected annotator underlined, seen in Fig.3.7.

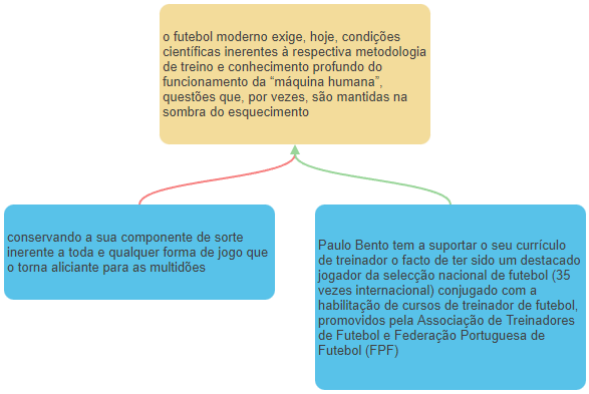

Figure 3.8: Example of the argument map component

3.4.1.D ArgumentMap Component

As seen in the requirements section, the framework needs to allow the analysis of the argument structure, making it possible to identify the present ADUs and relations. As the related work indicates, this analysis is possible through the use of argument map. Thus, this component is crucial to the framework.

Similar to the text, this component was already present in the framework, but was improved to use new characteristics of the opinion articles dataset, such as the types of ADUs and how the data was being processed, since the annotations were organized in a different way in the real dataset.

As the previous version of this component, a D3.js plugin, d3-flextree [3] was used, where each proposition is represented as a tree node and the relations between the annotations are represented as colored arrows.

In Fig.3.8, it is presented an example of a argument map developed by the VA Framework. This argument map shows three ADUs as rectangles of different colors, which correspond to their types: yellow for "Valor" and blue for "Facto". Between the conclusion and the left premise there is an attack relation (red arrow) and between the conclusion and the right premise there is a supporting relation (green arrow).

This component can be zoomed and panned if the user wants to visualize the map better, since some maps are more complex than others and might occupy more space.

3.4.1.E Network Graph Component

Comparing people, places and words is very common in this type of visualization to analyse the purpose of an argument. As seen in the state of art chapter, network graphs are frequently used to represent the connections between entities or keywords, depending on the data they are conveying.

Since the provided data present no information about the places and the people, the comparison of

[3] d3-flextree: https://github.com/Klortho/d3-flextree

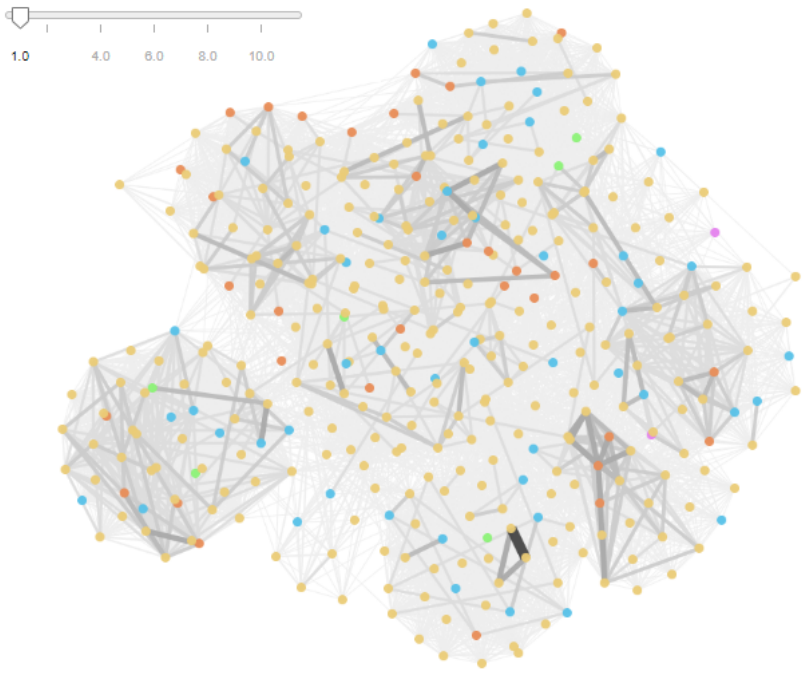

Figure 3.9: Example of the network graph component using Cytoscape.js

the documents can only be about the words that relate to the article or debate, depending on the dataset. Therefore, the connections will represent the similarity of the documents. To create this component, it is possible to have in mind the Collocates Graph tool that is part of the Voyant tools.

This component was already created but suffered huge changes since the information that it was conveying was completely different. The network graph is making use of a D3.js force model algorithm[4]. However, this algorithm was not creating good results for the opinion articles dataset, so another solution had to be chosen for the Dashboard platform. The solution was to use a JavaScript (JS) library called Cytoscape.js[5]. This solution, as well as the problems that led to it, will be further discussed on the following section, regarding the Dashboard platform.

In Fig.3.9, it is presented the network graph component using Cytoscape.js and in Fig.3.10, it is presented the network graph component using D3.js force model algorithm. Although they use different tools for the nodes layout, the behaviour is the same. In both representations, the documents correspond to nodes and the connections correspond to links. The stronger the link, the stronger the connection. In other words, if a link is darker and thicker, the documents connected to that link are more similar.

To navigate through the graph easily, a slider is presented next to the component, which filters the links according to their strength. By hovering over each node, a tooltip appears with information about the document. Each node can also be clicked to select a document, which will make the border of the node and its links black.

[4]force simulation: https://d3-graph-gallery.com/graph/network_basic.html
[5]Cytoscape.js: https://js.cytoscape.org/

Figure 3.10: Example of the network graph component using D3.js

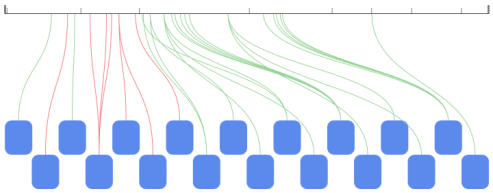

Figure 3.11: Example of the distribution component using the paragraphs in the axis

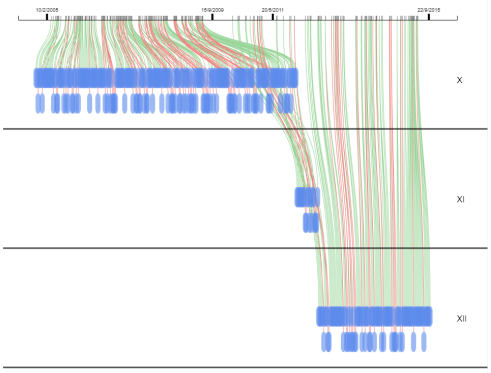

Figure 3.12: Example of the distribution component using the time in the axis

This component can be filtered, together with the List Component, by using the Filter Component. The documents that do not match the filters are faded away. Similarly to the Argument Map Component, the Network Graph Component allows zoom and pan for better navigation.

3.4.1.F Distribution Component

One aspect that must be allowed by the framework, that was inferred by the gathered requirements and the state of art, is the ability of seeing the documents through time or, in a more atomic scale, the annotations throughout the document.

To see this, one of the visualizations presented in the related work section, ToPIN, was used as inspiration. Similarly to the ToPIN tool, this component consists in a axis, rectangles and links that connect the previous two.

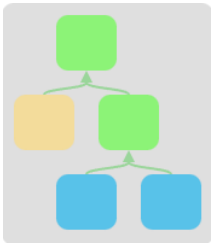

Figure 3.13: Example of the map overview component

In Fig.3.11, the component presents an axis that represents the length of the document. Each line separating the axis represents a paragraph. Since this component relates to the opinion articles dataset, the rectangles correspond to argument maps found on the document and the links correspond to the relations present in the respective argument map. The color indicates if it is a supporting (green) or attack (red) relation. The rectangles are not all in the same level so they do not occupy so much space.

In Fig.3.12, the axis represents time. Since this component is integrated in the ScrollyTeller platform, the rectangles represent initiatives. They are separated by legislature and according to their voting results, for better readability. The links are also colored according to their voting results. This component allows zooming in and out as well as panning, for a better navigation.

3.4.1.G MapOverview Component

As seen before, the Argument Map Component is an essential part of the framework. However, the previous version of the VA Framework did not allow to analyse the structure of more than one argument map without scrolling up and down multiple times. This could become stressful to the user. Thus, finding a way to show the structure of multiple argument maps easily and rapidly was important.

In the explored works, showing the only the structure or the full argument map were found but never together. However, join them makes the analysis of the overall structure of the arguments easier by having small representations of argument maps and then, if wanted, have a zoomed-in view of the argument map with the text.

The solution is presented in Fig.3.13, where it is possible to see an argument map without its text. This allows the user to observe the structure of the text, using the colors to identify the present ADUs and relations, without having to click on the argument map itself and be distracted by the text.

This component can then be accessible to the user so they can see the structure of more than one argument map without taking too long or needing too much memory.

3.4.1.H BarChart Component

In the state of art, both text visualizations and storytelling visualizations resorted to simple charts to show fundamental information about the data they were conveying. It is not illogical to think that charts like bar charts, stacked bar charts, line charts, etc., can be useful to show data that are present in the provided

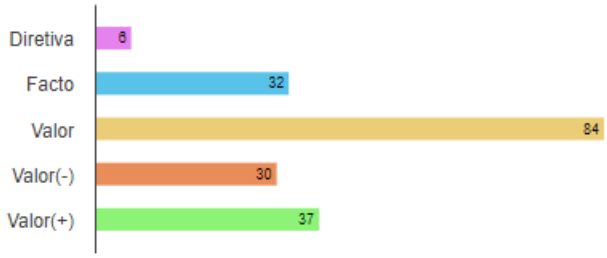

Figure 3.14: Example of the bar chart component

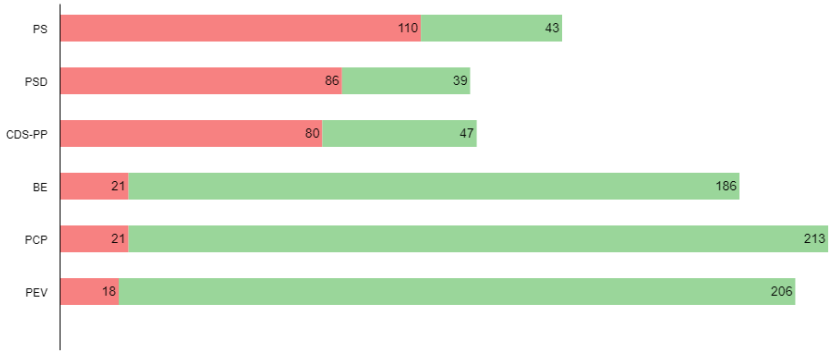

Figure 3.15: Example of the stacked bar chart component

datasets. Some of the requirements can be met with one of this straightforward charts. Although they are simple, the framework can improve it, by allowing interaction and customization.

A example of this new component can be seen in Fig.3.14. The names are on the left and the respective bars appear on the right, showing a number relating to the quantity of the elements that are being shown. In this case, the order of the elements is alphabetical, but the framework also allows it to be by the quantity. The example presents bars with different colors, but it also possible to use the same colors for all.

3.4.1.I StackedBarChart Component

A stacked bar chart is another example of a simple chart, that can be used to convey useful information. In the example shown in Fig.3.15, the structure is similar to the Bar Chart Component, except for the two series, red and green, that constitute the bar. Each color has a meaning depending on the data that is being conveyed. In this case, the bars uses numerical values, but the framework allows to percentages, creating equal sized bars.

3.4.1.J PositionChart Component

This component is a variation of the Stacked Bar Chart Component. While in the previous component it is already known by how many pieces the bar will be separated before hand, in this component it will be dynamic, according to the selected documents.

Figure 3.16: Example of the position chart component

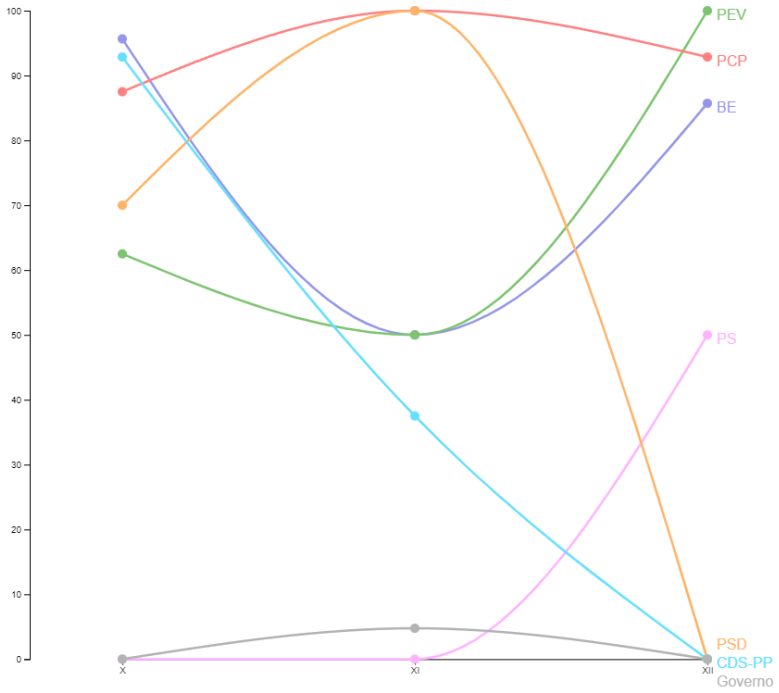

Figure 3.17: Example of the line chart component

The Position Chart Component was only used for the opinion articles. The component has two bars, separated by white lines; each piece represents a selected document. In Fig.3.16, the example shows the component with three documents selected. When hovering over on the first bar, a tooltip appears showing information about the document as well as the number respective to the bar.

3.4.1.K LineChart Component

The Line Chart Component is fundamentally a simple line chart that allows to see the evolution of anything through time. In Fig.3.17, there are two axis, the vertical axis, with the percentage values, and the horizontal axis, with time values – in this case, referring to legislatures.

The framework allows multiple series – different lines – with different colors, and writes a label for each one at the end of the respective line. For every point in time, a circle appears with the same color,

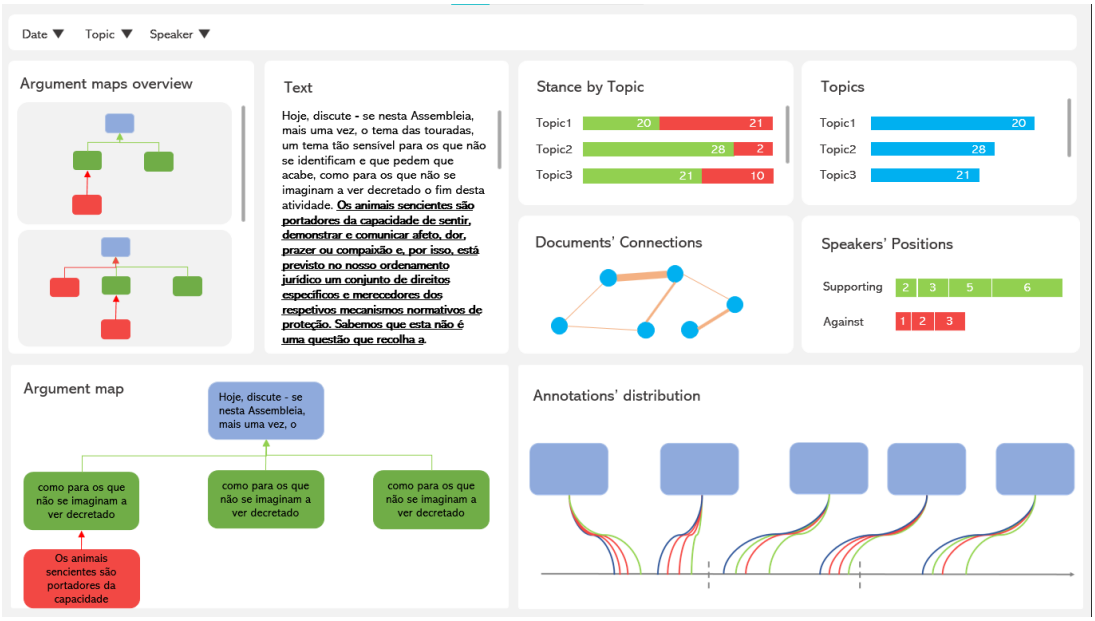

Figure 3.18: Dashboard Interface Prototype

for better readability of the chart. The labels are placed in a way so they do not overlap.

3.5 Dashboard

The first demonstrator to be developed was the Dashboard, which uses data about opinion articles. In the following section, the evolution of its development will be presented, starting from the first interface prototype, without any code, and then proceeding to the attempts of the platform creation with coding.

3.5.1 First Idea

The first interface prototype was created having in mind the related work seen in the state of the art section, as well as the requirements regarding the opinion articles dataset. The interface prototype is shown in Fig.3.18, which shows nine visual components.

At the top there is a filter bar, that serves as a filter to all the components present in the Dashboard, so it possible to filter the documents by date, topic and author.

As previously said, analysing the structure of more than one argument map easily is important. For that, there is a component on the left side of the prototype. In this prototype, unlike the previous version of the VA Framework, it is only possible to see the details of an argument map at a time, by selecting it in the Arguments Map Overview compartment. Both in the overview and in the argument map component, the argument boxes are colored according to their type. If it is a conclusion node then it's blue, otherwise it's red or green in case it is a node related to an attack or support relation, respectively.

Similarly to what already existed, there is a text component that shows the raw text of the articles, which match the applied filters, joined together.

On the top-right side of the prototype, four charts are presented. The network graph represents the connections between the different filtered articles. This component allows to associate the documents with each other, which was lacking from the previous version of the VA Framework.

The other three charts are a bar chart, a stacked bar chart and a position chart. The stacked bar chart allows to spot the topics with more attack or support relations. The bar chart shows the number of arguments for each topic. These last two components reflect how the visualization can provide topic analysis, which is, as seen in various works, very relevant to understand an argument.

In the VA Framework, there was a component where it was possible to see how many arguments a speaker had been against or supportive of. However, this component was not intuitive enough to see how many arguments there were in total for each relation. Considering this, a new visual element was introduced with two bars, one for the supporting relation, other for the against relation. Each have pieces that represent each author (which corresponds to an article) and each piece shows the number of associated arguments. This component is important to the platform because it allows the analysis of stance, which is a key characteristic of arguments.

Lastly, on the bottom-right side of the prototype there is a component that shows, mainly, the annotations distributions dispersed along the documents. The axis is separated in lines that represent the end of a document. Each rectangle represents a conclusion and by clicking on it, the corresponding argument map is shown. The connections between the rectangles and the timeline correspond to the relations present in that map, each with the color associated with their type. This component can provide a lot of information by itself, but it mainly allows time analysis.

3.5.2 First Attempt

After establishing the structure of the Dashboard, the coding of the platform took place. The coded prototype is similar to the interface prototype, since there was not yet feedback from the DARGMINTS project team. Therefore, changes that occurred that differ this prototype from the first one were result of some reflection while developing the structure.

The first aspect that was contemplated was the fact that this dataset carried more than 300 articles, which meant a complex navigation between them. Therefore, the way a user selects the documents they want to see should be facilitated. As seen in Fig.3.19, a list on the left side of the Dashboard was the solution for this problem. This allowed the user to see a list of all articles and select the ones they wanted to see, by clicking on the respective rectangle. The list also provided means to select all documents and remove all selections as well as search the list by the title of a document.

The rest of the platform remains with the same components seen in the interface prototype. Unfortunately, in this version, the Annotations' Distribution was not yet finished so it is not included in the screenshot. Similarly, the filter bar was not yet developed, which was supposed to be contained in the

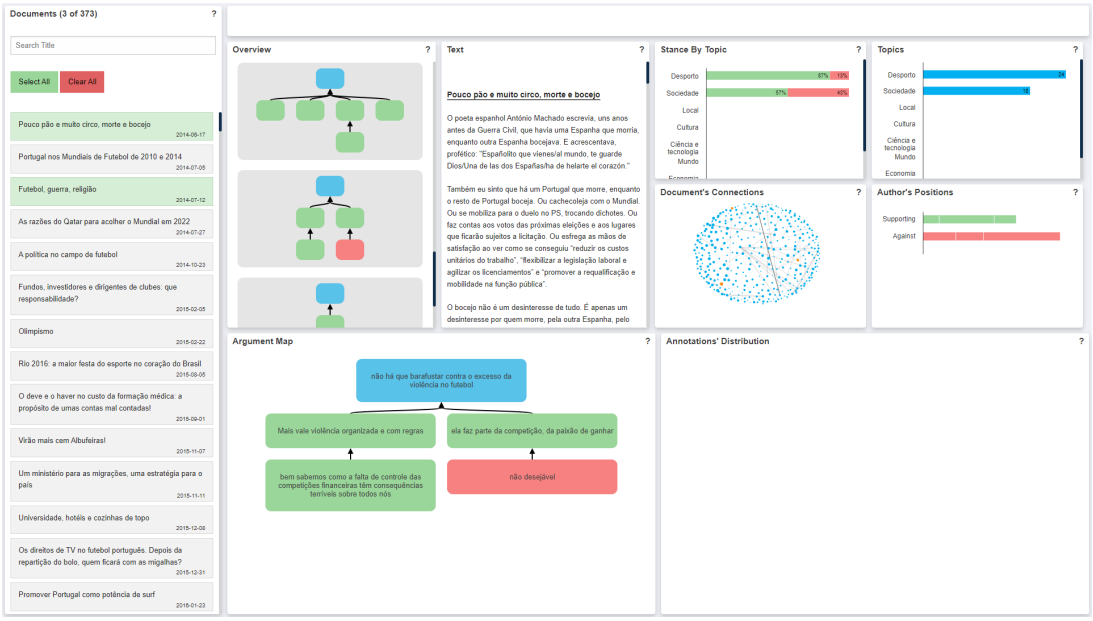

Figure 3.19: Screenshot of the first prototype of the Dashboard

top rectangle.

Regarding the integration of the opinion articles dataset, there was a complication concerning the argument maps, since these were in a different structure in the dataset than the framework was expecting.

In Fig.3.20 it is shown an example of an argument map in the structure provided by the dataset. This picture is from a tool, called ArgMine[6], that includes a drag-and-drop interface, which allows to annotate arguments from text and create this argument maps.

The different map layouts caused some conflict in the framework, since it was not prepared for this form of data. As it is possible to see in Fig.3.20, the argument maps in the dataset were composed by extra nodes that indicated the nature of the link between two nodes. Meaning that between two information nodes (blue) there was a support (green) or attack (red) node, with no text, that solely indicated the relation between those two nodes.

This varies from the expected layout, where the arguments maps are constituted by nodes with the information necessary to know if they are of support and attack, and where they connect directly to each other without extra nodes in the middle.

Therefore, the framework, mainly, this component, had to be adapted, which required some complex changes to translate from one layout to other. In particular, it was necessary to pass the information from the extra nodes to the information nodes, to have the desired layout.

The remaining components did not have problems with data integration, except for the map overview, which is similar to the argument map component.

[6] ArgMine: https://web.fe.up.pt/~argmine/

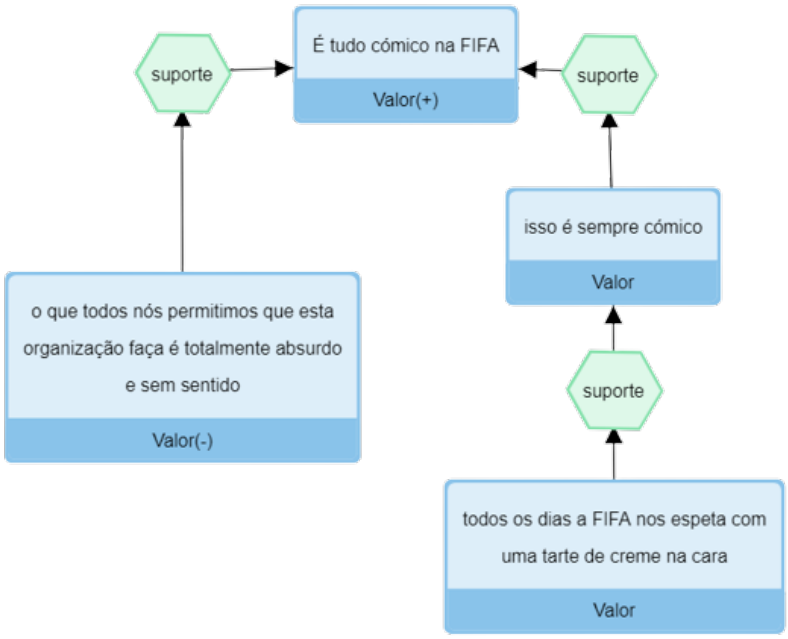

Figure 3.20: Screenshot of argument map in ArgMine

3.5.3 Prototype Changes

After obtaining feedback from the DARGMINTS project team, it was possible to understand the information they considered to be more important and needed to be conveyed and easily accessible.

The types of ADUs, for example, were not being conveyed in any way in the Dashboard, although it was an information provided in the dataset. This aspect was integrated in the argument maps, as seen in Fig.3.21, where the rectangles have the color corresponding to a specific type of ADU. The attack and supporting relations are conveyed in the arrows' color, instead of being depicted in the rectangles.

Another received comment concerned the component map overview, which they considered to be occupying to much space. Although the purpose of the component was to easily compare argument map structures, this was not enough to rob space from more important features. To come to a compromise, this component was integrated in the distribution component.

Therefore, when hovering on a blue rectangle, in the distribution component, the overview of the map will appear, as a tooltip, which will give an idea of the types of ADUs and relations that are present in the map, as seen in Fig.3.21. To accommodate this change, the components were redistributed to occupy the screen.

The network graph was also discussed, since it was not giving significant results due to the D3.js force model algorithm. As it can be observed in Fig.3.22, this algorithm provides various parameters to adjust the force that is applied to the nodes as well as to the links, such as the strength and distance variables, which are explained in comments in the figure.

Consequently, to respond to the recurring comments in the DARGMINTS meetings, these parame-

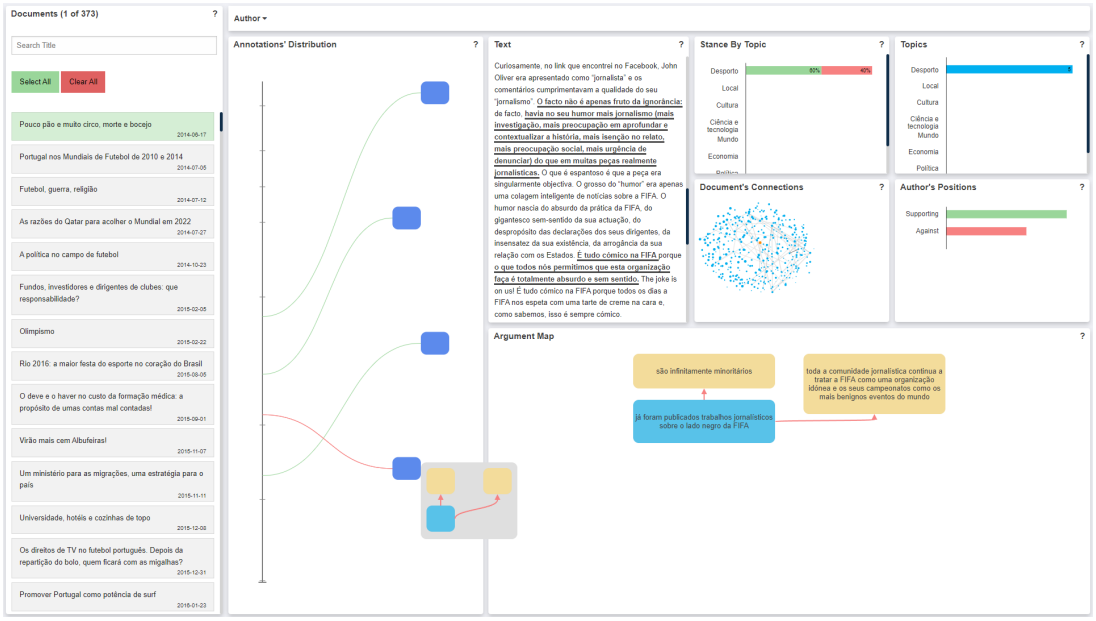

Figure 3.21: Screenshot of the second prototype of the Dashboard

```
var graphLayout = d3.forceSimulation(this.nodes)
    .force("charge", d3.forceManyBody()) // used to create the attraction/repulsion
    .force("center", d3.forceCenter(width / 2, height / 2)) // creates a new centering force with the specified x- and y- coordinates
    .force("x", d3.forceX()) // creates a new positioning force along the x-axis towards the given position x
    .force("y", d3.forceY()) // creates a new positioning force along the y-axis towards the given position y
    .force("link", d3.forceLink(this.links)
        .id(d => d.id) // sets the node id accessor to the specified function
        .distance(1000) // sets the distance accessor for each link to the specified number
        .strength(d => 1 / Math.min(count(d.source), count(d.target))) // sets the strength accessor for each link to the specified number
    )
    .on("tick", ticked) // called whenever a iteration occurs in the graph
```

Figure 3.22: Example of a D3.js force model creation

ters were changed, more than once, in order to create a stronger attraction to nodes with higher connections. However, small differences were seen regarding the network graph, since it still presented long strong links, which meant that documents with a higher number of common keywords were not closer from each other, thus increasing the difficulty in reading this graph.

After these changes, there were weekly meetings with the DARGMINTS project's team where it was possible to collect more opinions and important information to improve the platform.

The first obvious change when looking at Fig.3.23 is that the top and bottom of the right side of the platform swapped, making the argument map more noticeable.

However, the important changes occurred in the charts below. The bar chart that showed the number of relations by topic no longer exists and, instead, the Stance By Topic chart shows numeric values rather than percentage ones. This allowed the merging of two charts in one, giving space for new components and permitting important information to be conveyed. A new bar chart was created showing information about the types of ADUs, which is considered crucial for the DARGMINTS project team.

Despite the fact that many problems were addressed in these changes, various concerns remained. Some components, that were considered fundamental, and thus should be easily accessible and read-

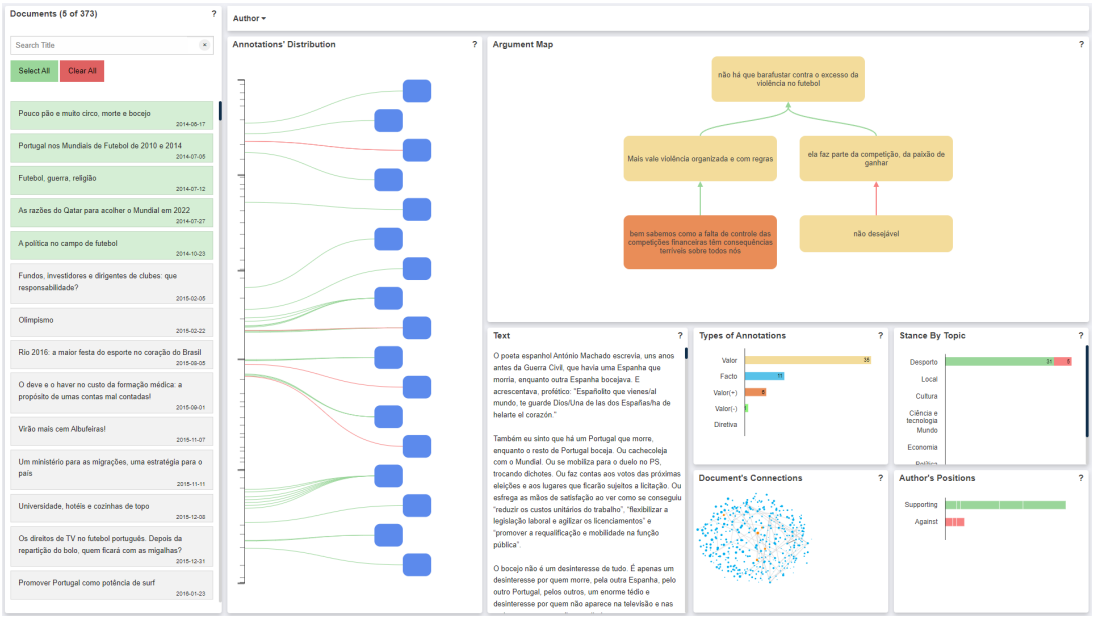

Figure 3.23: Screenshot of the third prototype of the Dashboard

able, were considered too small, such as the text component and the network graph.

The team also expressed that they would likely never want to compare more than two or three articles simultaneously. Therefore, the structure of the list allowing such selection and the fact the platform is designed for viewing all articles at once was not exactly what they were expecting.

Besides that, the dataset also presents information about the annotators. However, the way these entities should be conveyed was not yet clear, although it was important.

3.5.4 Final Approach

After more meetings with the DARGMINTS project's team, it was possible to gather more ideas and establish a structure that was suitable to resolve the problems and concerns encountered in the previous attempts. The new structure consists in two views: a global and a specific view, shown as tabs, both constituted by compartments that lodge the visual components created by the VA Framework.

The global view, as seen in Fig.3.24, presents the list component on the left, showing the multiple documents on the dataset. However, this component includes more filters, such as the date and the topic. This list is also divided in filtered documents and selected documents, to easily detected the made selections and facilitate the removal of such.

On the right side, in the top row, there's the network graph component showing the connections between these documents according to their keywords. Since this view has fewer components, the network graph was able to be larger and more readable.

However, as seen in the picture and in previous attempts, this component was not giving good results,

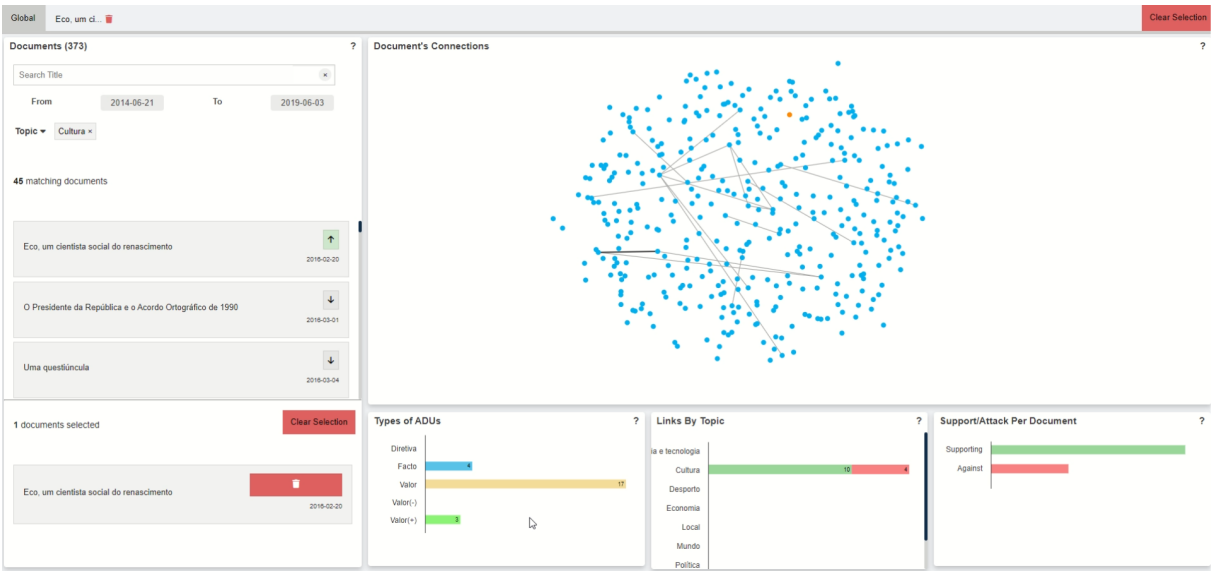

Figure 3.24: Screenshot of the global view of the Dashboard

as it was not possible to take significant conclusions about the connections of the documents. A solution for this problem will be discussed next.

Concluding this view, in the bottom row, three similar charts are presented: the bar chart, the stacked bar chart and the position chart. This charts are the same that the previous prototype had.

The specific view appears when a document is selected in the list and it only refers to that specific document. As seen in Fig.3.25, this view consists in three components with a group of buttons on the top-right side of the screen.

The group of buttons refers to the annotators of the selected article. The buttons 1 to 3 correspond to the annotators 1 to 3, while the button ”T” corresponds to all the annotators – ”Total”. By clicking on one of these buttons, it is possible to see the annotations of a specific annotator or of all of them. This affects both this view and the global one.

On the left side of this view, the distribution component shows the distribution of relations throughout the document. This component works as explained in the previous prototype, except it only shows the annotations for the selected document.

The full text of the document can also be analysed. This component, as explained in the previous section, changes according to the option selected in the group of buttons. If all annotators are selected, the text will present a heatmap showing the most argumentative parts of text. If only one annotator is selected, the text will show the annotations underlined with the color of the respective ADU.

As previously noted, the network graph, when using data about opinion articles, was not giving significant results. The D3.js force model algorithm did not seem to be applying the force necessary to approximate the most similar nodes as can be seen by the long strong lines present in the graph, as seen in Fig.3.24. Even after altering some values in this tool’s configuration, the structure of the network

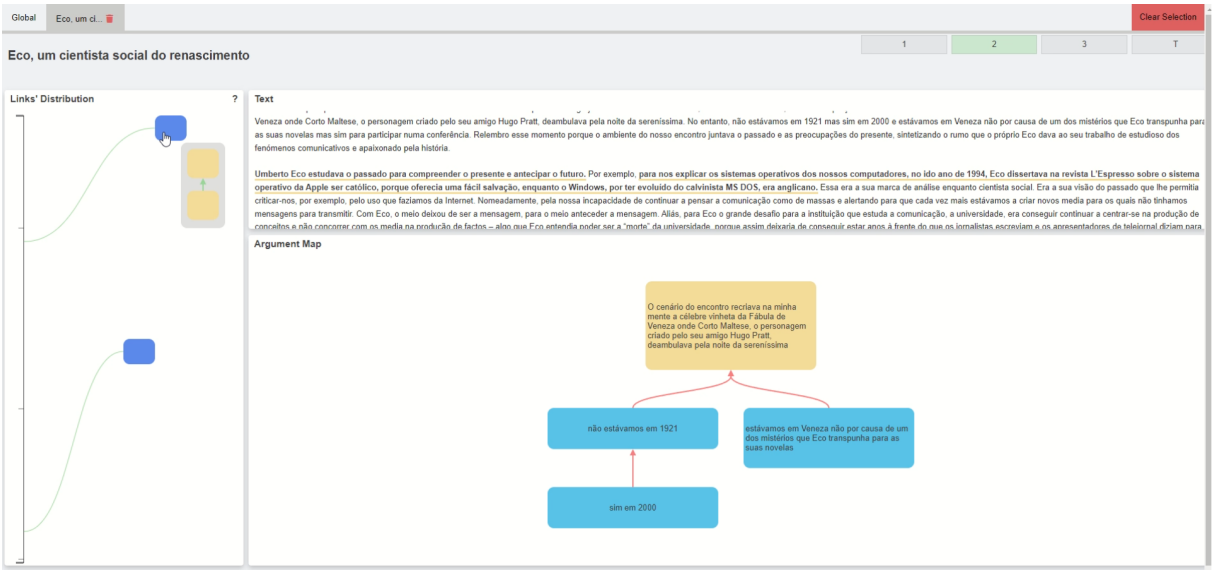

Figure 3.25: Screenshot of the specific view of the Dashboard

graph did not change.

Thus, other solutions were explored regarding the construction of network graphs. Cytoscape.js was one of the tools found that could improve the structure of the graph, while being easily integrated with the already existing libraries in the project. Cytoscape.js[7] is a JS library used for graph analysis and visualization that allows users to easily display and manipulate rich and interactive network graphs.

This library granted the same interaction level as D3.js, which makes it possible to use any of the libraries to create a graph, and allow the user to apply the same interaction mechanisms.

After integrating this library within the framework, the graph using data from opinion articles had more important results. As seen in Fig.3.9, there are notable clusters representing articles that depict similar subjects. This is due to the force applied on the nodes – when the link is stronger, the nodes are closer to each other. The use of this library, however, has a downside: the loading time, which is significantly longer than when using D3.js.

3.6 ScrollyTeller

The ScrollyTeller was the second demonstrator to be created, and uses data about parliamentary debates from 2005 to 2015, which corresponds to legislatures X, XI and XII. In this section, to better understand this platform, a brief explanation of the ScrollyTelling structure applied will be presented, followed by a segment describing the library used to create such structure. The remaining parts of the section describe the specific story the platform is conveying and then shows the final prototype.

[7] Cytoscape.js: https://js.cytoscape.org/

3.6.1 ScrollyTeller Structure

ScrollyTelling visualizations, as observed in the state of the art chapter, are a type of text visualizations that take a piece of data and use it to tell a story in an interactive and appealing way. As expected, they resort to visual components to show the data but, contrary to other visualizations, they are accompanied by a significant amount of text to direct the telling of the story.

This prototype focuses on a specific story while allowing the users some interactivity mid-narrative, so it is possible to explore the data before moving on to the rest of the story. Relating this to the three approaches seen in the state of the art chapter, the platform corresponds to a Interactive Slideshow. Furthermore, for the more curious users, an area is provided to allow the exploration of data that is not at all present in the story. This constitutes the structure of the ScrollyTeller demonstrator. To achieve it, a library was used in conjunction with the VA Framework, which will be discussed next.

3.6.2 Technology

To create the ScrollyTeller platform, a library with the same name was used[8]. The ScrollyTeller library is a JavaScript library that creates the HTML elements that allow the scrolling of data, which is retrieved from CSV files. By using this library, the story was separated into multiple sections, each with different graphs and narrations.

The library also allowed to link JSON triggers to events dispatched when each narration section comes into view, which permits the changing of the graphs of each section while the user is scrolling and the story is being told. Therefore, the usage of the ScrollyTeller library allowed a development more focused on the visual components of the VA Framework and their integration with another type of visualization than with HTML elements and scrolling events.

3.6.3 Story

A ScrollyTelling visualization, as previously said, needs a story to tell its readers. Analysing the data about parliamentary debates, it was possible to propose an interesting story. In order to analyse the provided datasets, Flourish[9] was used, a tool that can generate instantaneous charts to observe interesting patterns in the data and quickly take conclusions and form stories.

Since the data relates to the legislatures X, XI and XII, it is possible to compare the three, once they faced entirely different circumstances regarding their govern. The first had absolute majority of Partido Socialista (PS), while the second one had relative majority of the same party. The last legislature changed govern, with relative majority of Partido Social Democrata (PSD).

[8] ScrollyTeller: https://github.com/ihmeuw/ScrollyTeller
[9] Flourish:https://app.flourish.studio/

Figure 3.26: Screenshot of the initial page of the ScrollyTeller

This comparison allowed the analysis of patterns and the evolution of the parties' initiatives along the years. After some examination, using the different charts provided by Flourish, evidence that initiatives got more rejected in the first legislature became clearer. Therefore, the premise of the story turned into if absolute majorities indicated more rejected initiatives or not.

To convey this, the story was divided in parts. The first part relates to a general view of all the legislatures, where all the initiatives are shown, the rejected and approved ones by party and how the parties usually vote aligned with others, without presenting distinction between legislatures. Contrary to this, the second part presents the data separated by legislatures. The next part combines the previous ones, indicating in a line chart the percentage of rejected initiatives throughout the legislatures by party.

The last part regards to the conclusion of the story, which answers to the premise of the story. The end of the story confirms that the absolute majority, associated to legislature X, caused a bigger percentage of rejected initiatives.

3.6.4 First Idea

Having in mind the chosen structure, library and story, it was possible to start idealizing what the ScrollyTeller platform would look like. A list of the visual components that would be present in the story was done, describing the interaction and animation they would provide. Flourish was used to create the multiple visual components in order to get graphs similar to the ones the solution would have. This list served as a guide in the development process of the final prototype.

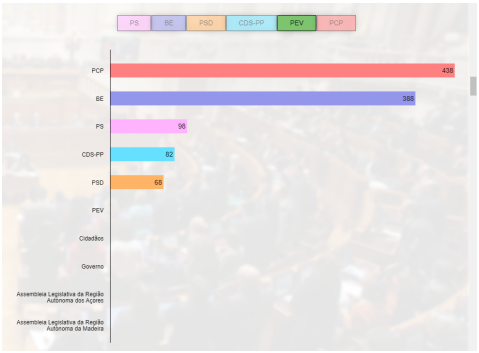

Figure 3.27: Screenshot of the bar chart component with buttons, with option ”PEV”

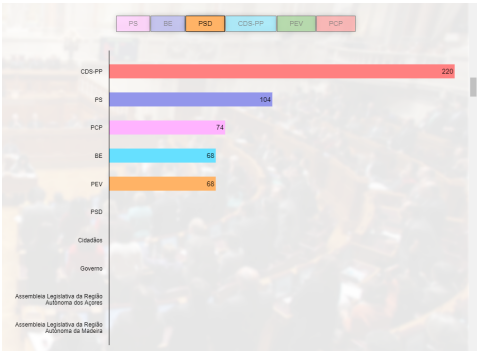

Figure 3.28: Screenshot of the bar chart component with buttons, with option ”PSD”

Figure 3.29: Screenshot of the stacked bar chart component before scroll

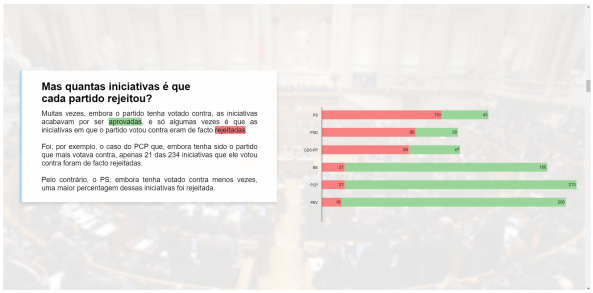

Figure 3.30: Screenshot of the stacked bar chart component after scroll

3.6.5 Final Prototype

After establishing the structure of the story, the development of the prototype for the ScrollyTeller platform started. The construction of this demonstrator was significantly easier and faster than the first. This relates to the fact that by this time the VA Framework was already improved, because, when the Dashboard was developed, the framework was changed. Due to its versatility, the framework was easily adapted to accommodate another dataset, in order to create the ScrollyTeller.

Nevertheless, due to the fact the data was divided in multiple files, it was harder to pass it to the platform in the format they were in. To address this problem, the files were processed, using Python[10] scripts. These scripts were able to create new datasets with the needed fields from two different datasets or with completely new computed fields that were later used by the platform. This processing was done previous to the development of the platform to save time in its loading.

The difficulties regarding the development of this platform itself concerned external aspects, such as the used library. Every time an unknown library is used, a certain amount of time is necessary to learn how to work with it. Specific features of this tool were more complex to understand, which slowed the development time. Another struggle when creating the ScrollyTeller was writing the story. Since it portrays a complex subject, the story needs to be written with care and preciseness, which requires time

[10] Pyhton https://www.python.org/

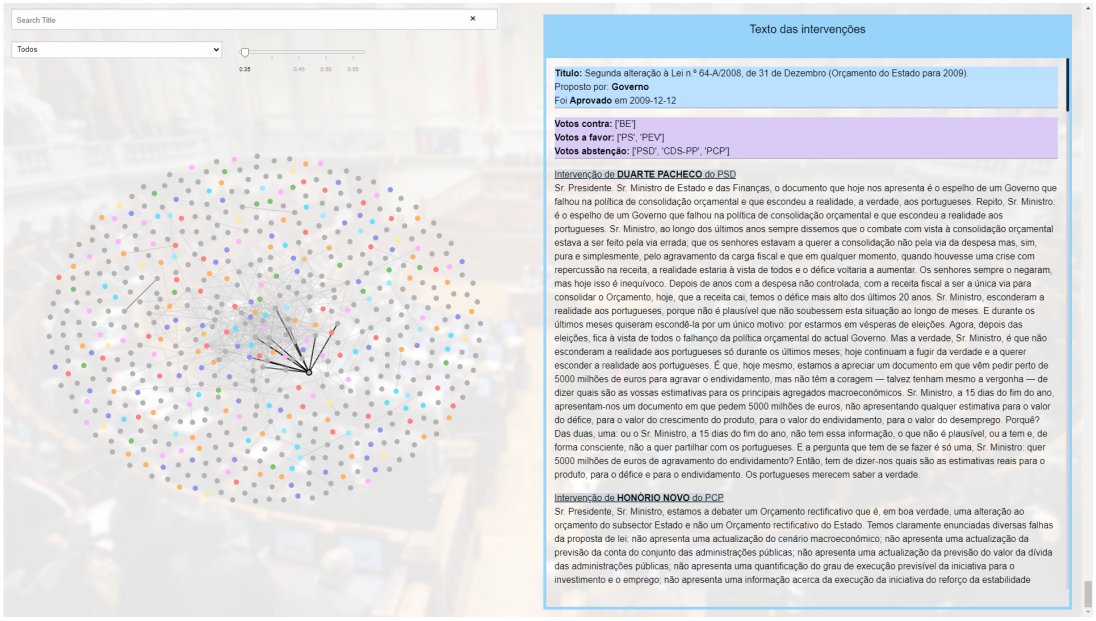

Figure 3.31: Screenshot of the exploratory area of the ScrollyTeller

and political knowledge.

This demonstrator starts with the initial page, stating the title of the story, which can be observed in Fig.3.26. Afterwards, it is presented a brief introduction about the data used for the story, in order to contextualize the user about it. Following this, the platform maintains the same pattern: container of text on the left accompanied with a graph on the right. Most of the visual components that were used in the Dashboard demonstrator were used in the ScrollyTeller, such as the bar chart and the stacked bar chart, which can be seen in Fig.3.27. The distribution component was also used but it was adapted, since it was conveying more data. For this platform, it was created a line chart to convey the necessary data.

Some components changed through scroll, as allowed by the library, seen in Fig.3.29 and Fig.3.30, and others changed through buttons, which is the case of the presented bar chart, in Fig.3.27 and Fig.3.28. As previously described, this platform includes an exploratory area, similar to the Dashboard demonstrator, which consists mainly in a network graph and a text box.

This area can be seen in Fig.3.31, which presents, on top, a series of filters that can be applied to the network graph below. This graph presents all the initiatives, in a force model algorithm created by D3.js. Contrary to what happen in the Dashboard, this force model algorithm had good results with this dataset and it was not necessary to use the Cytoscape.js library. When a node is clicked on the graph, it is highlighted, and the details regarding that initiative, such as the voting results and the debate interventions, appear on the right side. This area allows the user to know all the information of all the initiatives present in the dataset.

4

Evaluation

Contents

4.1 Tests with Non-Experts 56
4.2 Tests with Experts 65
4.3 Discussion 69
4.4 Improvements 69

In order to evaluate the developed platforms and confirm if the requirements, specified in the previous chapter, were met, the testing plan was divided in two phases: tests with non-experts and tests with experts. The first phase corresponds to a usability testing, where the platforms are tested regarding its usability by people who are not experts in the platform's domain, which, in this case, is linguistics. In the second phase, the tests are conducted with people from the DARGMINTS project, which are experts in the platform's domain. This will give different and more precise feedback, regarding the platform's features.

Tests with programmers were not conducted to evaluate the VA Framework, since they had already been done previously for the previous version. Furthermore, the focus of this work is conveying how versatile the framework can be through the creation of the two demonstrators.

This chapter explains in detail both testing phases, discussing their results. Following this, it is listed the improvements made on each platform.

4.1 Tests with Non-Experts

As previously noted, in the first phase, the purpose was to evaluate the usability of the platforms with people outside of the DARGMINTS project. All tests were conducted in person, in Portuguese language, and took place between September 19, 2022 and October 2, 2022. The guide and protocol for this testing phase can be found in Appendix B.

4.1.1 Users Sample

As can be read in the guide and protocol, the users are presented to a survey in the beginning of the test, which accompanies them throughout the whole test. After the user consents to the usage of their data for statistic means, where their data will be anonymous, the users are asked to answer questions of demographic nature.

Twenty people took part in this testing phase, 15 women and 5 men, which are dispersed through various age groups, as can be observed in Fig.4.1. The majority of them have a bachelor's degree, most in engineering, the others have a secondary level of education.

The next section of the survey regards the interpretation of five charts to understand the knowledge of the users on that subject. For each chart, there is a multiple choice question the user must answer. The charts shown are a bar chart, a pie chart, a line chart, a network graph and a box plot. In Fig.4.2, it can be seen that most users answered correctly to all questions, and the question regarding the network graph was the one that had more incorrect answers. Besides that, only one user answered incorrectly to two questions, all others gave the wrong answer to only one question, which can be a sign of lack of attention. After filling the initial sections of the survey, the users were explained how the tests were

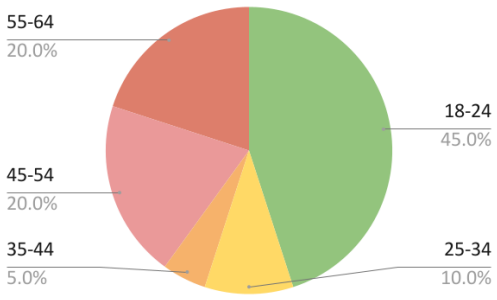

Figure 4.1: Age groups of the users of the first testing phase

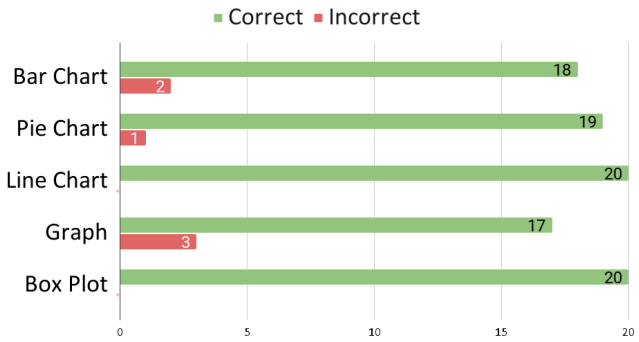

Figure 4.2: Correct and incorrect answers for the charts section of the initial survey

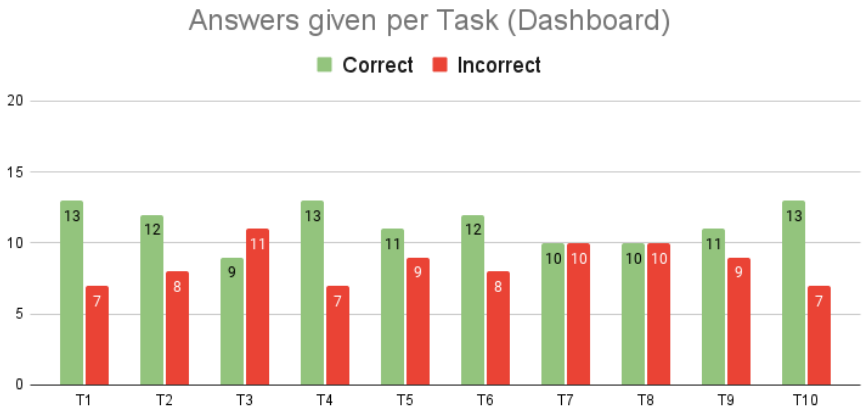

Figure 4.3: Correct and incorrect answers for the Dashboard tasks

going to proceed.

4.1.2 Dashboard Testing

The first demonstrator to be tested was the Dashboard. In order to familiarize the users to the concepts used in this platform, such as arguments, ADUs and annotators, a video explaining these, as well as the general structure of the platform, was played to the users in the beginning. Following this, the Dashboard was presented and the user had no more than five minutes to freely explore the platform.

After this, the user had to perform ten tasks, with different levels of complexity, related to all the visual components present in the platform. The tasks were performed in a random order to minimize learning effects. Every task started in the same state – no documents or filters selected. For every task, time was measured and the user was asked to answer a NASA Task Load Index (NASA-TLX) [21] survey, which consists in measuring six aspects on a scale from 1 to 7, in order to calculate an overall workload score for each task, in other words, the general effort the user felt performing each task.

In general, the users showed some difficulties performing the tasks, as seen in Fig.4.3, by the signif-

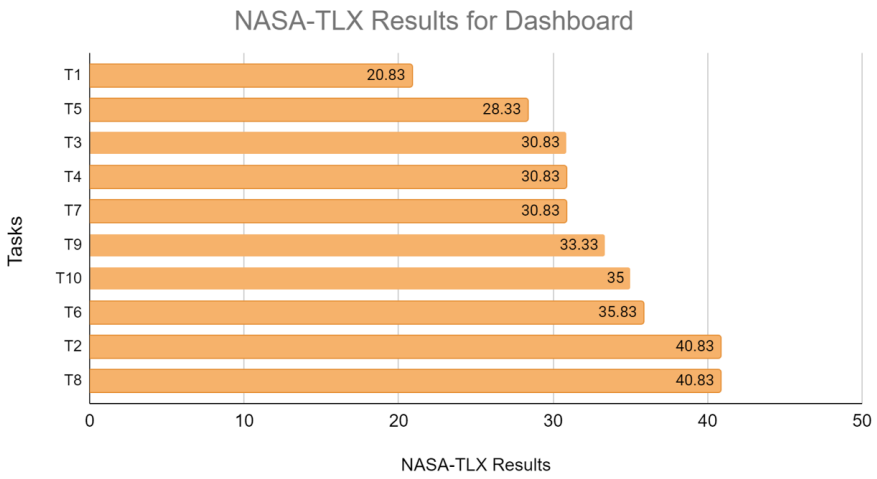

Figure 4.4: NASA-TLX survey scores for the Dashboard tasks in the first testing phase

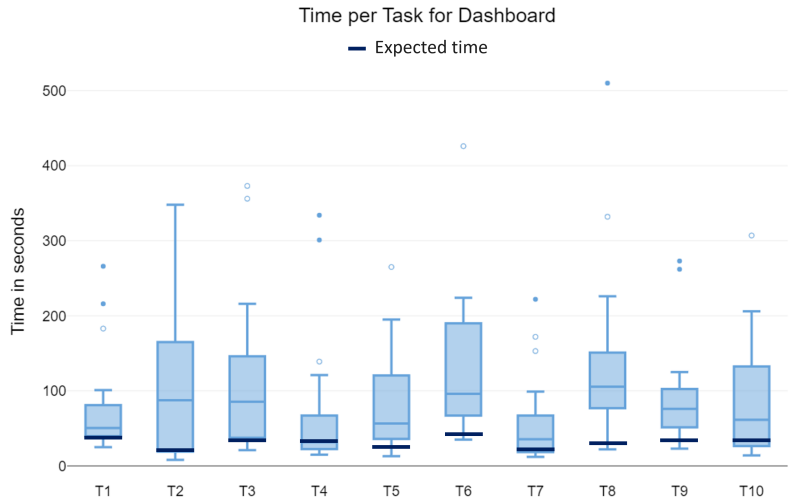

Figure 4.5: Time spent on each task of the Dashboard

icant number of incorrect answers in all tasks, as well as the rather high effort they felt performing them, as can be observed in Fig.4.4. The time spent on each task related to the Dashboard, as well as the expected times can be observed in Fig.4.5.

During the tests, it was possible to pinpoint some core aspects of the platform that caused more confusion and frustration between the users. The rest of the section is divided in segments that detail these aspects. The last segment explains the System Usability Scale (SUS) [22] survey results, filled at the end of the Dashboard testing.

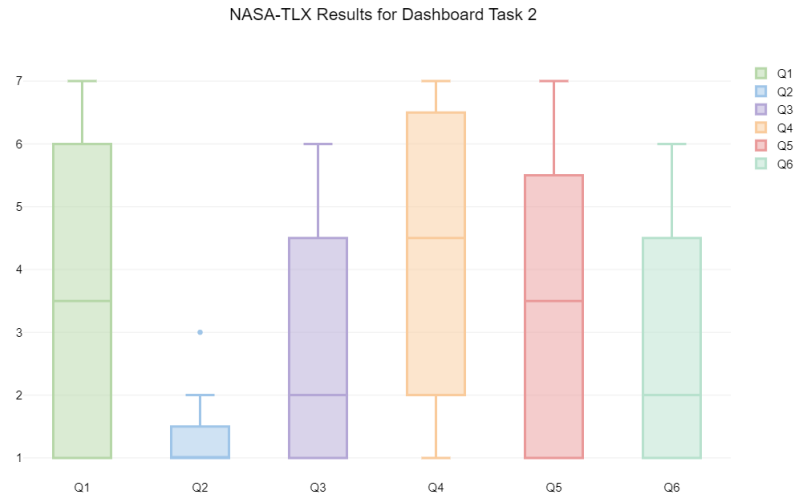

Figure 4.6: Dashboard's NASA-TLX survey results for Task 2

4.1.2.A Network Graph

The Network Graph component revealed to be one of the aspects that users felt more lost at, shown by tasks 2 and 3, which asked to find the documents with the stronger connections. One of the tasks had the extra step of filtering the documents by a specific topic first. Task 3 received more incorrect answers than all other tasks while task 2 had an result of 40.83 on the NASA-TLX survey, which it is already considered a bit of an effort. The details of these results can be seen in Fig.4.6, which presents a box plot for each of the NASA-TLX questions.

For the users, this component was not intuitive enough to rapidly acknowledge what it was conveying. Although the title described "Document's Connections" they did not understand what the connections were actually portraying. This was due to the lack of a caption or a guide explaining the component and how the present documents were being connected.

Besides that, the library used for this graph structure, Cytoscape.js, allowed nodes to overlap, so some nodes ended up being hidden by others. In some cases, the users were unable to complete the task or answer correctly due to this characteristic.

In task 3, where users where asked to first filter the documents by a topic, the slider, included in the network graph component, should be used to aid the navigation. However, the slider, not carrying any caption, was not noticeable enough and users did not understand its purpose. This was made worse by the fact that they perceived connections as integers and the slider's steps were decimal.

When users understood the slider's purpose, another aspect would make them hesitate: the slider would not update according to the filtered documents. Instead, it would allow the user to slide to the maximum number of connections of all documents, although the maximum of the filtered ones was less. This was not intuitive to the users, and confused a lot of them in this task, which led them to incorrect

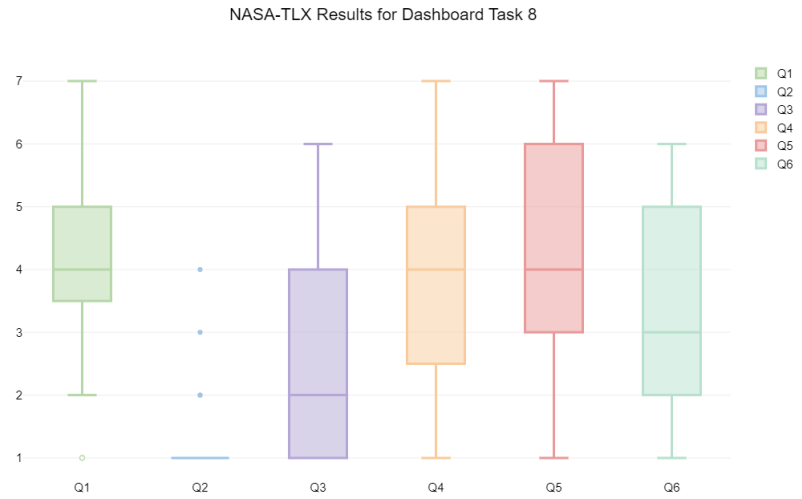

Figure 4.7: Dashboard's NASA-TLX survey results for Task 8

answers.

All these aspects in the network graph component led to a significant increase in time spent performing the tasks, especially task 2. The average time spent performing this task was 1 minute and 49 seconds, while the expected time was 20 seconds, seen in Fig.4.5.

4.1.2.B Annotators

Since users were people outside the DARGMINTS project, they were unfamiliar with the concept "annotators" and how they are responsible for registering the arguments present in text. This concept was, of course, explained in the video at the beginning of the test. However, when the tasks regarded the analysis of annotations of a specific annotator the users became confused and could not point where the platform was conveying information relating this.

Tasks 4 and 5 required the selection of a certain annotator in the corresponding group of buttons. Still, a lot of users would not notice them. When they were clicked, the users did not understand the differences they were observing. This was especially seen when they navigated to the global tab and did not notice that the charts had updated with the selected annotator.

Nonetheless, task 8 was the one most users felt most difficulty with, which is demonstrated by the charts in Fig.4.3 and Fig.4.4, that show this task received many wrong answers and proved to be a challenge to the users. In this task the users were asked to point the paragraph the annotators found more arguments in common, however, users were not able to understand that the colors in the text represented just that, since they associated those colors to other entities present in the platform, instead of an intensity heatmap. Overall, the mental demand of this task revealed to be, on average, medium (value of 4), which can be seen in the box plot presented in Fig.4.7. To take in these struggles, the users

spent a lot more time performing this task, taking an average of 2 minutes and 17 seconds, represented in Fig.4.5.

4.1.2.C Other Problems

As previously noted, users felt difficulties performing most tasks. A recurring problem that a lot of users went through was the incorrect filtering of the documents by a topic. Many users would not notice the button to filter by topic and would instead search the topic on the search bar, which is solely for titles. This, would of course lead to incorrect answers.

Another problem, seen in a lot of tests, was the inability to associate the types of ADUs and the types of relations to the given colors. This prevented the users to comprehend the structure of the argument maps. Some of the correct answers in the tasks that relate to this, such as tasks 5, 9 and 10, were achieved because the users resorted to what they remembered from the video.

4.1.2.D Special Cases

Since there was a reasonable quantity of users, and they were disperse in various age groups, it was possible to collect data from diverse perspectives, mainly people with special eye conditions.

From the 20 users, two had presbyopia, which is farsightedness caused by loss of elasticity of the lens of the eye, that usually occurs in middle and old age. The users with this condition had extra difficulty interpreting certain aspects of the platform, such as the date in the documents list, due to the small font-size. Besides that, one of them did not notice the scroll bar because it was too thin.

Another one of the users was color blind, which prevent them to distinguish certain colors. In this case, yellow, green and red were very identical to them.

4.1.2.E SUS

As previously said, at the end of the Dashboard testing, the users were asked to fill a SUS survey, to evaluate the platform in terms of its usability, where they classify ten sentences based on their level of agreement, on a scale from 1 to 10.

As can be observed in Fig.4.8, the results of this survey present some divergence in the users' opinions regarding the Dashboard. Users had various struggles when performing the tasks, which were aggravated by the unfamiliarity of the concepts the platform was conveying, such as annotators, arguments and ADUs.

This created some complexity for the users when navigating the platform, which made them feel they were not prepared for such tool. This is seen by answers given in questions 4 and 10, which regard the need of technical support or of learning a lot of things to be able to use the platform.

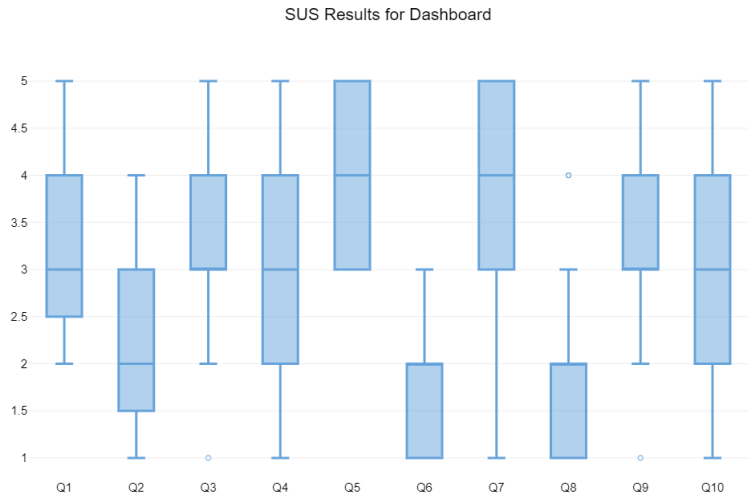

Figure 4.8: Dashboard's SUS survey results in the first testing phase

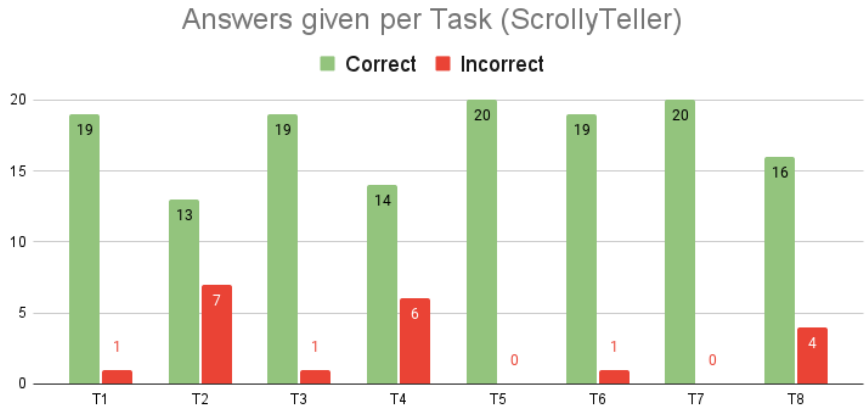

Figure 4.9: Correct and incorrect answers for the ScrollyTeller tasks

Nonetheless, the users found the platform's functionalities were well integrated without many inconsistencies. Most users did not think the platform was unnecessarily complex and thought it was easy to use.

Overall, the users disagree when it comes to use this platform frequently. While some saw the advantages this platform could bring to the linguistics area and how it could be used with other data, others did not see it as something they would ever use, since it was not of their particular area of interest.

Finally, the SUS survey allows the calculation of a score, between 0 and 100. For the Dashboard testing, the SUS score was 62.9, which is considered below average. In other words, on a scale from A+ to F, it is a C-. This is strongly related to the difficulties the users felt during the tasks.

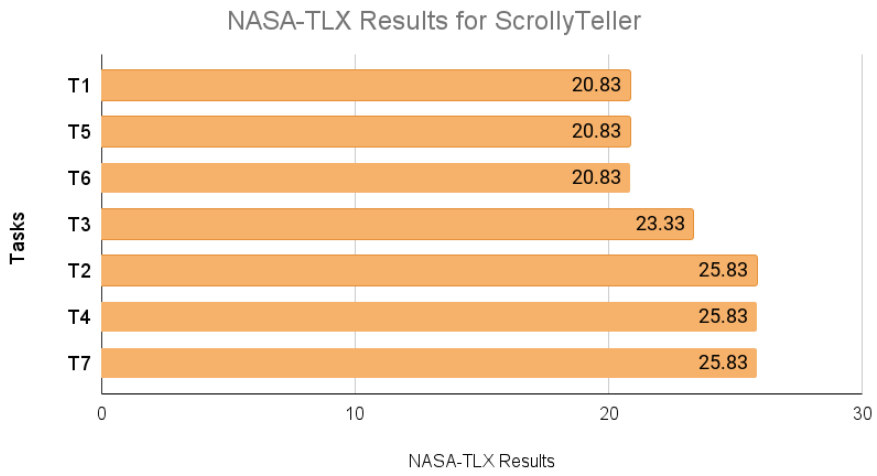

Figure 4.10: NASA-TLX survey scores for the ScrollyTeller tasks in the first testing phase

4.1.3 ScrollyTeller Testing

After finishing the Dashboard testing, a brief demonstration of the ScrollyTeller was done, explaining the core points of its structure. Following this, the users were given a few minutes to read the full story to contextualize them about the data.

Afterwards, the users were asked to answer to eight questions, in a similar way to the Dashboard testing. As previously, for every task, they had to fill a NASA-TLX survey. All questions can be found in the guide and protocol in Appendix B.

Contrary to the Dashboard, the users found this platform more intuitive and easier to navigate, which can be seen by the lower number of incorrect answers in Fig.4.9 and the low score results in Fig.4.10. Relevant details and the SUS survey results will be discuss next.

4.1.3.A Writing

The distinguishing characteristic of this platform is the fact that it combines text with visual components to tell a compelling story to the user. When one of these aspects fails to convey the data, the story is no longer being transmitted as clear and precise as it could be.

While testing the ScrollyTeller, many users faced struggles because of how the story was being told in terms of Portuguese. Whether it was minimal orthographic errors, lack of punctuation or lack of precise terms to make the sentence more understandable, the users felt distracted and frustrated when reading the story, which meant it was not being conveyed correctly.

4.1.3.B Components

If the visual components are not conveying the data intuitively and precisely, then the story also fails to be told correctly.

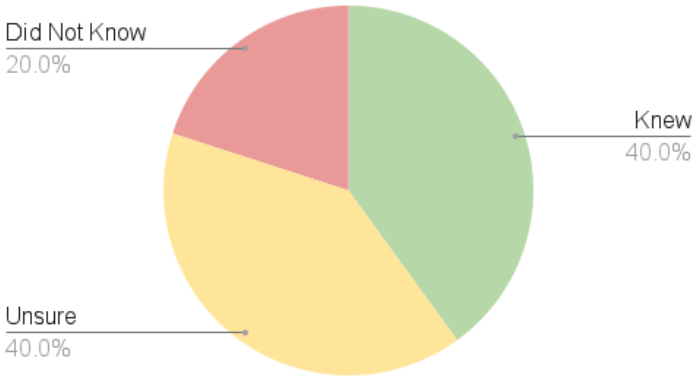

Figure 4.11: ScrollyTeller Task 8 answers

In task 3, the users had to scroll down until they reached the line chart and a percentage that responded to their question appeared. However, they only scrolled down to the line chart, without that percentage. To get that number they tried clicking on the respective node on the chart, expecting a tooltip with that information.

In task 7, the users had to go to the exploratory area and search an initiative. Although, most of the users acknowledged what they had to do, after writing the name on the search bar they did not realize the graph had updated. This got them frustrated and made them spent a lot more time in this task, taking an average of 1 minute and 45 seconds.

4.1.3.C Overall Story

The purpose of the last task was to see if the users understood the story, mainly its conclusion. As can be seen in Fig.4.11, when asked if there was a higher percentage in rejected initiatives in legislature X than in the remaining legislatures, 40% of the users said yes, while other 40% said yes, but was unsure about it and had to check in the platform. The rest of the users did not know the answer.

4.1.3.D SUS

Similarly to the Dashboard testing, at the end of the ScrollyTeller testing, the users were asked to fill a SUS survey. The results can be seen in Fig.4.12, which, contrary to the first test, show more convergence between users.

Overall, the users felt that the platform was easy to use and it was not unnecessarily complex. They thought the ScrollyTeller functionalities were well integrated without many inconsistencies.

Although most users answered they would not need technical support or need to learn anything to be able to use the platform, some users still felt lost navigating it, probably due to the unfamiliar structure.

In general, most users would like to use this platform frequently (average of 4, which represents agree), although some disagree with this statement. However, this can be due to the fact that the

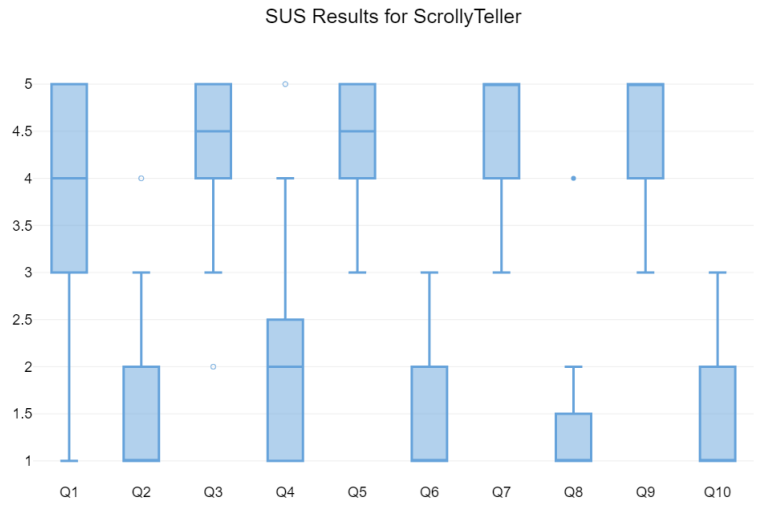

Figure 4.12: ScrollyTeller's SUS survey results in the first testing phase

ScrollyTeller is a storytelling visualization and not a consultation platform, which means users would probably visit it once and not frequently as the question states. That thought can send the user to a different interpretation of the question, and thus a lower answer value. For the ScrollyTeller, the SUS score was 83.9, which is above average, and corresponds to an A, on a scale from A+ to F.

4.2 Tests with Experts

The second phase of testing focused on the DARGMINTS project's requirements and if the features of the platforms, mainly the used visual components and their interactions, were fulfilling them. Contrary to the other phase, these tests were conducted online, through Zoom[1], since in person meetings were complicated due to the users' living locations. Zoom allows to remote control the host's computer, which, in this case, is great to create a controlled environment for all tests. This testing phase took place between October 7, 2022 and October 14, 2022. Its guide and protocol can be found in Appendix C.

4.2.1 Users Sample

Similarly to the first stage of testing, the users are accompanied by a survey throughout the test. After consenting to the usage of data, they are presented with a section about their demographic data. Five people from the DARGMINTS project performed this test, all of them men. Three of them had a doctorate degree and the remaining ones had a master's degree.

In this survey, it was also asked how often they analysed argumentative text. One person answered they do it every day, two answered they do it many times and the remaining two answered sometimes.

[1] Zoom: https://zoom.us/

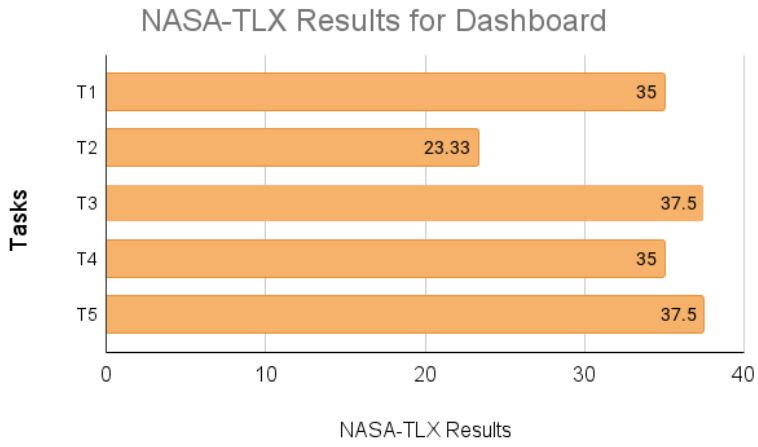

Figure 4.13: NASA-TLX survey scores for the Dashboard tasks in the second testing phase

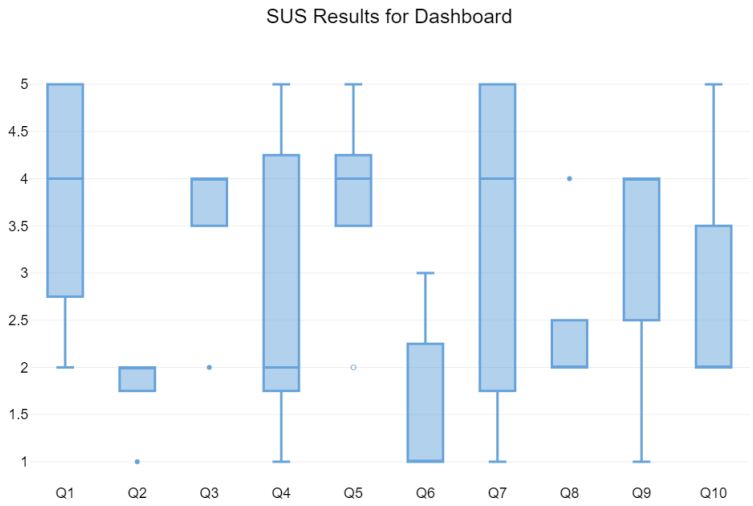

Figure 4.14: SUS survey results for the Dashboard in the second testing phase

Following the same structure of the first phase, the users were presented with the same charts and questions. In this testing phase, everyone answered correctly to every question.

4.2.2 Dashboard Testing

Since this testing phase was conducted with people from the DARGMINTS project, it was not necessary to play a video explaining some of the concepts present in the platform. Instead, the test proceeded to the platform's demonstration right away. Afterwards, the users were given 5 minutes to freely explore the Dashboard, while being encouraged to say what they were thinking, a method called "think aloud".

After that, the users were asked to perform five tasks, which were the ones the non-experts found the most difficulties, using the "think aloud" method. Similarly to the first phase of testing, the users had

to fill a NASA-TLX survey for each task. The tasks can be found in the protocol in Appendix C.

Overall, the users answered correctly to all tasks, except task 5, where 2 users answered incorrectly. This is due to the previously discussed problem about the difficulty of associating the types of ADUs and relations to the given colors. This task was also the one that presented the higher value of effort in the NASA-TLX survey, seen in Fig.4.13.

Throughout the tests, it was possible to collect a lot of feedback using the ”think aloud” method. The users in this testing phase, which were experts in this area, explained they understood how a non-expert could be lost in this platform due to the complexity of the concepts related to argumentation. Because of this, they stated the platform needed to be more intuitive than ever, to aid the users who are not familiar with these terms to navigate through the Dashboard.

Most of the received comments were discussed in the first phase of testing with non-experts. However, these users had feedback that went beyond usability improvements. Related to the difficulty the non-experts felt regarding the argumentation concepts, they suggested the platform should have a form of tutorial in the beginning with arrows pointing to the components and text boxes explaining what the component was conveying. This would break the shock non-experts have when coming in contact with complex concepts they do not know nothing about.

When asked to fill the SUS survey, some were concerned most people would need help to be able to use this platform, which can be seen be the lower values answered in question 7. For them, however, the platform felt easy to use and its functionalities seemed well integrated, without a lot of inconsistencies. The lower values given in the SUS survey, seen in Fig.4.14 were result of problems discussed in the previous testing phase, which were not so hard to surpass for these users, since they did not have the extra difficulty of not understanding the concepts in the platform. This resulted in a SUS score of 66, which corresponds to a C in a scale from A+ to F.

4.2.3 ScrollyTeller Testing

Identical to the previous testing phase, a brief demonstration of the ScrollyTeller took place after the end of the Dashboard testing. After the users read the full story, taking their time, and using the ”think aloud” method once again, they were asked to perform six tasks.

The difficulties these users faced were seen in the first stage of testing, with non-experts. Some of these struggles were, as previously discussed, related to inconsistencies in the writing of the story. Another aspect these users discussed was the fact of getting frustrated by having to scroll up and down to see a certain part of the story. To address this, they suggested having an index that could accompany them through the story and allow them to jump from part to part. This allows users to only see the parts of the story they want. Consequently, an index could make ScrollyTeller be seen as a consultation platform instead of a website article that they only visit once.

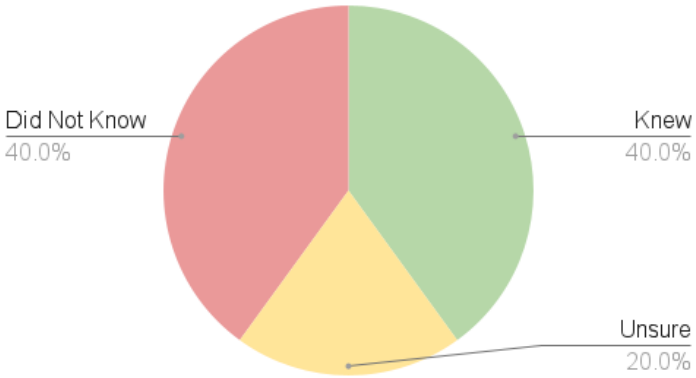

Figure 4.15: ScrollyTeller Task 6 answers

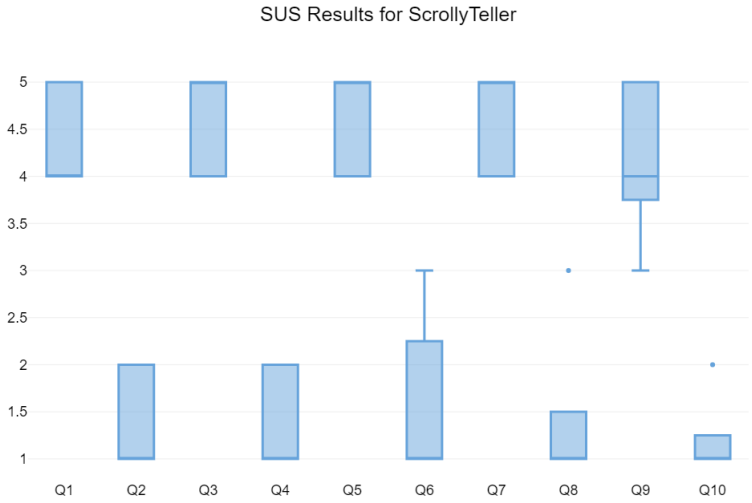

Figure 4.16: SUS survey results for the ScrollyTeller in the second testing phase

The last task serves to see if the users understood the story correctly. As seen in Fig.4.15, 40% of the users knew that there was a higher percentage of rejected initiatives in legislature X than the remaining ones, while 20% were not sure about and had to check this information in the platform. The rest of the users did not know the answer.

As seen in Fig.4.16, the users agreed the platform was easy to use and its functionalities were well integrated, without many inconsistencies, besides the writing ones. They stated this platform was more accessible to other people, since it did not address complex concepts related to argumentation. The SUS result for the ScrollyTeller in this phase was 88.5, which corresponds to A+, the highest grade in a scale from A+ to F.

4.3 Discussion

Comparing the first and second phase of testing, it is possible to observe the platforms had similar results, according to the difficulties the users faced when navigating through them.

Regarding the Dashboard, although the experts in the area of linguistics were able to find the answers they were looking for more easily than the others, they still pointed out the same problems present in the demonstrator the remaining users did, which would be aggravated by the lack of familiarity of the concepts related to argumentation.

These problems were mainly related to the absence of captions that would aid the users associate the given colors to their corresponding concepts as well as explain certain aspects of some visual components. Because of this lack of captions, the users felt more confused and frustrated when using this demonstrator.

Contrary to this, the ScrollyTeller was seen as a more accessible and easy platform, both by non-experts and experts. Since it does not require previous knowledge of complex concepts about argumentation, this demonstrator is more straightforward.

The biggest struggle the users faced was inconsistencies seen in the writing of the story. However, the majority of the users, in both testing phases, understood the story, and its conclusion, although some were unsure and checked the information. Overall, this platform had better results than the Dashboard, due to its lower complexity.

The performed tests showed how versatile the framework can be. Although using the same visual components created by the VA Framework, it was possible to create two completely distinct platforms that had different impacts and opinions from the users that tested them. Therefore, this evaluation proved the versatility of the framework, which was the purpose of creating two demonstrators.

4.4 Improvements

Using the feedback collected from both testing phases, it was possible to make a list of improvements for the two platforms.

4.4.1 Dashboard

The new version of the Dashboard can be found in the following link: `https://web.ist.utl.pt/ist189407/DARGMINTS/`, where the improvements that will be discussed next can be seen.

To answer to the comments received on both testing phases concerning the Dashboard, a caption explaining the colors given and how they were associated to the types of ADUs and relations was created. Regarding the heatmap about the common arguments registered by annotators, the color

scale was changed to a monochromatic one and a caption explaining it was added, when all annotators were selected.

Although the previous version of the Dashboard already had a feature which provided pop-up boxes containing explanations about each component, the information was not updated. Considering this, in the new version these boxes carry updated and useful information about the charts, that aids the user navigate through the platform.

Another component that was significantly altered, due to the collected feedback, was the slider, which is now updated according to the filtered documents and only changes integer by integer.

While conducting the tests, some bugs of the platform would come to the surface. One of these was related to the date picker, which sometimes would not close after choosing a date. Besides that, the platform was letting the users choose dates beyond the dates of the most recent and oldest articles, which created confusion.

In the previous version of the VA Framework, the date picker was created by a JS library called dtsel.js[2], which provided a simple design and could be easily integrated with the platform. However, this library is very restricted and does not provide a parameter to prevent the selection of certain dates.

To bypass this, a new data picker library was found[3], which can deliver the same simple design and easy integration as the previous library, while providing a lot more features, such limiting the selection of dates. Other features were explored but not used, since they were not important to the Dashboard.

This demonstrator also received multiple aesthetic changes, to make the platform more pleasing as well as more noticeable for people with eye conditions. These alterations refer to, for instance, the rounding of the buttons and containers and the increase of font-size in the documents list.

4.4.2 ScrollyTeller

The new version of the ScrollyTeller can be found in the following link: https://scrollyteller.herokuapp.com/, with the improvements that will be discussed next.

As noted in the previous sections, the writing of the story was the most commented aspect in the ScrollyTeller testing. To improve this, each section of the story was reviewed and rewritten if needed, using better and more precise terms. To aid the transmission of the story to the reader, the text was edited with bold and colored excerpts to highlighted important statements or titles.

Concerning the visual components, a tooltip was added to the line chart component that allowed to see all the values shown in each legislature, by hovering over each node, which was already the behaviour the users were expecting from the platform during the tests.

Regarding the network graph, in the exploratory area, since the users showed some difficulty ac-

[2]dtsel.js: https://github.com/crossxcell99/dtsel
[3]vanillajs-datepicker: https://github.com/mymth/vanillajs-datepicker

knowledging the graph updated after the typing of the title in the search bar, both the graph and the search bar were altered.

When a title is searched, a list of suggested titles, compatible with the searched one, will appear. This helps the user focus in the wanted initiative. When the user searches one title, the graph will update as before, except the filtered nodes will have a black border so they are highlighted and more noticeable than before.

5

Conclusion

Contents

5.1 Limitations and Future Work 74

Despite the fact argumentation is an important aspect in our daily lives, analysing argumentative text is not a simple task, due to the complexity and density of arguments. This thesis, aligned with the DARGMINTS project, presented the objective of creating a solution that would allow both experts and non-experts from the linguistics field to easily understand and explore this type of text.

To achieve this, an already developed framework would have to be improved and adapted to real and complex data that referred to two different domains: opinion articles and parliamentary debates. In order to show these case studies, two web platforms would be created, using, for this, different structures, demonstrating the framework's versatility.

To create the two demonstrators, a search for visual representations to show the logical structure of arguments started. The most common, and still used nowadays, are argument maps, which are essentially concept maps, but the nodes are connected by logical links.

Since the visualizations surrounding argumentation are scarce, there was a need to turn the research elsewhere. Text visualizations provide a lot more insights in how to convey certain aspects that are present in argumentative text. A type of text visualization is called storytelling, which was also explored in the research, in order to find new creative ways of telling stories about the provided data.

From this research it was possible to notice some crucial elements that needed to be included in the new version of the VA Framework, such as the annotated text, arguments maps and network graphs. Beyond those, each platform would need an extra set of charts to convey its own data.

Having this in mind, it was possible to establish the requirements for both demonstrators. From there, the framework started to be altered, changing its architecture accordingly. The first demonstrator, the Dashboard, started to be created, using various approaches on the way, which resulted on a set of visual components, where users can easily analyse the argumentative text present in opinion articles. The next demonstrator was the ScrollyTeller, which resulted in a story about parliamentary debates using visual components created by the VA Framework. During all this development process, the DARGMINTS project team provided feedback about the platforms during weekly meetings, which helped guide the solution to its best outcome.

In order to evaluate the web platforms, there was a two-phased testing, which consisted in testing the demonstrators with both experts and non-experts in the linguists field. Overall, the Dashboard was perceived has a more complex platform because of its nature and structure while the ScrollyTeller felt more easy to navigate and less confusing because of its accessibility regarding the theme.

The Dashboard was altered mainly to be more intuitive, especially to non-experts whilst the ScrollyTeller received changes related to the writing style to make the story more understandable and precise.

5.1 Limitations and Future Work

With the information collected from the tests, it was possible to pinpoint some interactions or features that the users would like to see in both platforms.

Concerning the Dashboard, the users, especially the DARGMINTS project team members, were frustrated to see the various visual components could not be rearranged. During the tests, many were the testers that tried to resize the container of the text component. It would be interesting to create this feature in this platform to make a more personalized platform. Other interactions were missed by the users, such as a sorting feature in the documents' list or a way to select an ADU and show it in the text component.

Finally, another aspect that annoyed the users was constantly going to the opened tab every time they selected a document. A new mode of selecting documents could be explored, for instance, having two paths, one for just selecting without going to the tab and other for selecting and going to the tab.

The Dashboard was received as a complex platform mainly due to the data it is conveying. To circumvent this, a brief tutorial explaining the basic argumentation concepts and how the platform works would benefit inexperienced users, and prevent the feeling of confusion.

Regarding the ScrollyTeller, the users saw this platform more as an website article than as a consultation tool, since it consists in just telling a story from start to end. However, this opinion could be changed if an index was added to this platform. If this index allowed to go back and forth in the story and provided the current location of the user, the user would have more freedom about what to read in the story, and would not be forced to scroll down to the parts that interest them.

Bibliography

[1] H. Pashler, *Encyclopedia of the Mind*. SAGE, 2013.

[2] J. Oliveira, "Visual argument," Ph.D. dissertation, Instituto Superior Técnico, 2021.

[3] J. Lawrence and C. Reed, "Argument Mining: A Survey," *Computational Linguistics*, vol. 45, no. 4, pp. 765–818, 01 2020. [Online]. Available: https://doi.org/10.1162/coli_a_00364

[4] M. Davies, "Concept mapping, mind mapping and argument mapping: what are the differences and do they matter?" *Higher Education*, vol. 62, pp. 279–301, 2011.

[5] T. van Gelder, "Argument mapping with reason!able," *The American Philosophical Association Newsletter on Philosophy and Computers*, vol. 2, 2002.

[6] ——, "The rationale for rationale™," *Law, Probability and Risk*, vol. 6, no. 1-4, pp. 23–42, 2007. [Online]. Available: https://doi.org/10.1093/lpr/mgm032

[7] J. Lowrance, I. Harrison, A. Rodriguez, E. Yeh, T. Boyce, J. Murdock, J. Thomere, and K. Murray, *Template-Based Structured Argumentation*. London: Springer London, 2008, pp. 307–333. [Online]. Available: https://doi.org/10.1007/978-1-84800-149-7_15

[8] D. Kiesel, P. Riehmann, H. Wachsmuth, B. Stein, and B. Froehlich, "Visual analysis of argumentation in essays," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1139–1148, 2020.

[9] H. Prakken, S. van den Braak, H. van Oostendorp, and G. Vreeswijk, "A critical review of argument visualization tools: Do users become better reasoners?" in *Workshop Notes of the ECAI-06 Workshop on Computational Models of Natural Argument (CMNA-06)*, K. Reed, C. Grasso, F., Ed. ECCAI, 2006, pp. 67–75.

[10] C. Twardy, "Argument maps improve critical thinking," *Teaching Philosophy*, vol. 27, 06 2004.

[11] J. Anthony Blair, *Studies in Critical Thinking: 2nd Edition*. Windsor: WSIA, 2021, pp. 117–149. [Online]. Available: https://doi.org/10.22329/wsia.08.2019

[12] C.-Y. Sung, X.-Y. Huang, Y. Shen, F.-Y. Cherng, W.-C. Lin, and H.-C. Wang, "Topin: A visual analysis tool for time-anchored comments in online educational videos," in *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 2185–2191. [Online]. Available: https://doi.org/10.1145/2851581.2892327

[13] N. Cao, D. Gotz, J. Sun, Y.-R. Lin, and H. Qu, "Solarmap: Multifaceted visual analytics for topic exploration," in *2011 IEEE 11th International Conference on Data Mining*, 2011, pp. 101–110.

[14] J. Sheidin, J. Lanir, P. Bak, and T. Kuflik, "Time-ray maps: Visualization of spatial and temporal evolution of news stories," in *EuroVis 2017 - Short Papers*, B. Kozlikova, T. Schreck, and T. Wischgoll, Eds. The Eurographics Association, 2017.

[15] X. Zhang, S. Chandrasegaran, and K.-L. Ma, "Conceptscope: Organizing and visualizing knowledge in documents based on domain ontology," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–13.

[16] S. Chandrasegaran, S. K. Badam, L. Kisselburgh, K. Ramani, and N. Elmqvist, "Integrating visual analytics support for grounded theory practice in qualitative text analysis," in *Computer Graphics Forum*, vol. 36. Wiley Online Library, 2017, pp. 201–212.

[17] K. Kucher, R. Martins, C. Paradis, and A. Kerren, "Stancevis prime: visual analysis of sentiment and stance in social media texts," *Journal of Visualization*, vol. 23, 08 2020.

[18] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu, "StoryFlow: Tracking the evolution of stories," *IEEE Transactions on Visualization and Computer Graphics (Proceedings of IEEE InfoVis 2013*, vol. 19, no. 12, pp. 2436–2445, 2013.

[19] K. Watson, S. S. Sohn, S. Schriber, M. Gross, C. M. Muniz, and M. Kapadia, "Storyprint: An interactive visualization of stories," in *Proceedings of the 24th International Conference on Intelligent User Interfaces*, ser. IUI '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 303–311. [Online]. Available: https://doi.org/10.1145/3301275.3302302

[20] E. Segel and J. Heer, "Narrative visualization: Telling stories with data," *IEEE transactions on visualization and computer graphics*, vol. 16, no. 6, pp. 1139–1148, 2010.

[21] A. Cao, K. K. Chintamani, A. K. Pandya, and R. D. Ellis, "Nasa tlx: Software for assessing subjective mental workload," *Behavior Research Methods*, vol. 41, no. 1, pp. 113–117, Feb 2009. [Online]. Available: https://doi.org/10.3758/BRM.41.1.113

[22] J. Brooke, "Sus: A quick and dirty usability scale," *Usability Eval. Ind.*, vol. 189, 11 1995.

Datasets

Structures of the datasets of the two case studies.

A.1 Opinion Articles

Structure of a JSON file regarding the dataset of the opinion articles case study.

- **id** – article unique identifier to use at MongoDB;
- **title** – article title;
- **authors** – list of authors;
- **body** – article raw content;
- **url_canonical** – original article url;
- **publish_date** – article date of publication;
- **meta_description** – small paragraph containing a summary of the content of the document;
- **source** – source of the article; this field is always ”Público” since only those news articles were considered;
- **keywords** – article keywords list (obtained from the original article);
- **topics** – article topics, based on the list of keywords;
- **argument_annotations** – list of annotations; each one following the Argument Interchange Format (AIF) format;
  - **nodes** – nodes list, which represent the declarations, and their attributes;
    - **id** – node unique identifier;
    - **text** – annotated text;
    - **type** – either an information node, containing annotated text, or a relation node, which can be of two types: support relation or attack relation;
    - **label** – node classification type; if it is an information node then in can be Facto, Valor, Valor(-), Valor(+) or Diretiva;
    - **ranges** – char level indices of the annotated text from the article body;
  - **edges** – directed edges list, which represent the links between the declarations, and their attributes;
    - **from** – id of the source node;
    - **to** – id of the target node;
  - **metadata** – only used in the ArgMine Platform;
    - **annotatorId** – annotator unique identifier.

A.2 Parliamentary Debates

Structure of a CSV file regarding the dataset of the parliamentary debates case study.

- **ini_leg** – initiative legislature;
- **ini_id** – initiative id;
- **ini_num** – initiative number;
- **ini_type** – initiative type: ”Proposta de Lei” or ”Projeto de Lei”;
- **ini_title** – initiative title;
- **ini_session** – initiative session;
- **author** – initiative author;
- **date_start_leg** – legislature start date;
- **date_end_leg** – legislature end date;
- **vot_results** – initiative voting results;
- **vot_in_favour** – initiative votes in favour;
- **vot_against** – initiative votes against;
- **vot_abstention** – initiative abstention votes;
- **pub_num** – debate number;
- **pub_session** – debate session;
- **pub_date** – debate date;
- **pub_leg** – debate legislature;
- **pub_serie** – debate serie;
- **pages** – pages in the debate relating to the intervention;
- **pdf_file_path** – source path of the debate document;
- **doc_first_page** – debate document first page;
- **dep_id** – congressperson id;
- **dep_gp** – congressperson party;
- **text** – intervention text;

B

Testing Protocol – Non-Experts

Testing Protocol – Non-Experts

Required Material

For this test, the conductor must use the same computer with connection to the internet and access to the following links:

- Dashboard: https://web.ist.utl.pt/ist189407/DARGMINTS/
- ScrollyTeller: https://scrollyteller.herokuapp.com/
- Form: https://forms.gle/RhQuMgow8hvQURuX7

The conductor must also carry material to collect data gathered from the test, which includes pen and paper.

Testers

For this testing phase, it will be selected at least 20 non-experts.

Beginning of the test

All tests are presential. Although they are conducted in different places (most of them in INESC-ID) the same computer was used.

A form is presented to the tester at the beginning of the test, as they enter the room, with the following introduction:

> "Bem-vindo(a)!
>
> O meu nome é Ana Silva e estou a completar o mestrado em Engenharia Informática e de Computadores no Instituto Superior Técnico.
>
> Neste momento, estou na fase de testes de uma framework desenvolvida no contexto da minha dissertação, em conjunto com o projeto DARGMINTS.
>
> Essa framework chama-se Visual Argument e visa tornar mais fácil analisar texto argumentativo, usando componentes visuais interativas e intuitivas.
>
> Para mostrar a sua versatilidade, criei duas plataformas, o Dashboard e o ScrollyTeller, com diferentes estruturas mas ambas usando a mesma framework como base.
>
> Para as testar, gostaria que fizesse parte do processo de testagem, ao dedicar algum do seu tempo e atenção.

O teste consiste em realizar algumas (8 a 10) tarefas para cada plataforma, tendo algum tempo para explorar as plataformas e fazer perguntas antes.

Este teste será realizado no INESC-ID, na sala 533, e durará cerca de 1 hora, sendo possível desistir a qualquer momento sem qualquer penalização.

O teste será anónimo e todos os dados colecionados ao longo do teste serão apenas para efeitos estatísticos. Obrigada desde já pela sua atenção e contribuição para a avaliação deste projeto."

After they read the introduction they can consent to the read information when they ready and begin the tests.

Part 1 - Initial Survey

Before starting the tests themselves, the tester will have to fill some demographic questions, relating to their age, gender, visual difficulties and level of schooling.

Afterwards, the tester will be asked to analyze five charts: a bar chart, a pie chart, a line chart, a graph and a box plot. This will help to understand the level of knowledge of the testers regarding the analysis of charts.

Part 2 – Dashboard Testing

Part 2.1 – Demo and Exploration

Since the Dashboard analyzes the structure of arguments in a deep manner, some concepts need to be taught to the tester in order to facilitate the usage of the platform. Therefore, the tester will have to see a video explaining the structure of the platform as well as the necessary concepts to better understand it.

After watching the video, the tester will have exactly 5 minutes to freely explore the platform. They are encouraged to say out loud what they are thinking and feeling.

Part 2.2 – Tasks

Protocol

The tester will be asked to perform ten tasks for this platform. Each question will be said out loud and clarified if any questions are asked. The questions are showed in paper, highlighted with a sticker, and in a word document in the computer, so the tester can easily access it while doing the task. The

tester can also ask the researcher to repeat the task as many times as they want while doing the task.

The tester must answer the question out loud so the researcher know they have finished the task.

The order of the questions will be random (determined by a Latin Square) for all testers.

All questions begin with no documents and no filters selected.

The researcher must register if the tester was successful doing the task, as well as the time the tester took to finish the task.

After every task, the tester must answer a form (NASA-TLX survey) regarding the effort, satisfaction and frustration they felt performing the task.

Tasks Description

The following tasks are described in Portuguese.

Tarefa 1 – Quantos artigos sobre Desporto foram realizados entre 12 de Junho de 2014 e 18 de Dezembro de 2016?

Time: 40 s

Process: Go to filters on the left and choose “Desporto”. Then choose the dates “2014-06-12” and “2016-12-18” to filter the list of documents. OR choose the dates first and then the topic. Read the “matching documents” number.

Answer: 25

Objective: Understand if the user can use the filters and know the list changed. See if the “matching documents” number is visible.

Tarefa 2 – Quais os nomes dos documentos que têm uma conexão mais forte, ou seja, que têm mais palavras chaves em comum entre todos os documentos?

Time: 20 s

Process: Identify strongest line in the graph, which represents more keywords in common between documents. Hover on the nodes connected to that line and say their names out loud.

Answer: “Serralves, sintoma e panorama nacional” and “Quem pode, manda”

Objective: See if the user can analyze the graph.

Tarefa 3 – Quais os nomes dos documentos com o tópico Sociedade que têm uma conexão mais forte, ou seja, que têm mais palavras chave em comum?

Time: 35 s

Process: Select the topic “Sociedade”. Identify the strongest line in graph. Use slider to help the analysis. Hover on the nodes connected to that line and say their names out loud.

Answer: “Quo vadis obesidade” and “A obesidade como catástrofe ambiental dos tempos modernos”

Objective: See if the user can use the filters along with the slider of the graph.

Tarefa 4 – O anotador 3 que anotou o artigo “A mais recente receita para salvar o capitalismo” registou quantos mapas de argumento?

Time: 34 s

Process: Search document “A mais recente receita para salvar o capitalismo”. Click on it in list of documents to open its tab. Select annotator 3 and analyze “Link’s Distribution” chart. Count number of blue rectangles that represent the argument maps.

Answer: 5

Objective: Understand if the user acknowledges the “Link’s Distribution” chart as being the distribution of links, where each rectangle is a argument map. And if they understand that the chart varies with the selection of the annotator.

Tarefa 5 – Quantos ADUs do tipo Facto registou o anotador 2 no artigo de opinião “Depois de Salvador Sobral”?

Time: 25 s

Process: Search document “Depois de Salvador Sobral”. Click on it in list of documents to open its tab. Select annotator 2 and go to the “Global” tab. Analyze “Types of ADUs” chart and observe bar that correspond to the “Facto” ADU. See the number present in the bar.

Answer: 3

Objective: See if user can interpret the “Types of ADUs” chart”, understanding that it varies with the selection of the annotator.

Tarefa 6 – Entre o artigo mais antigo sobre Desporto e o mais recente, qual deles teve mais ligações de ataque? O mais antigo ou o mais recente?

Time: 45 s

Process: Select the topic “Desporto”. Select the oldest and newest articles in the list of documents, corresponding to the first (“Pouco pão e muito circo, morte e bocejo”) and last (“Jogador residente, o aplauso de uma raridade”) of the filtered list. Go to the “Global” tab and analyse the “Support/Attack Per

Document" chart. Observe the bars corresponding to the attack links and see which one is bigger. To see the numbers, it is possible to use the tooltip by hovering on the bars.

Answer: "Pouco pão e muito circo, morte e bocejo" (oldest)

Objective: Understand if the "Support/Attack Per Document" chart is intuitive and if the user understands they can answer the question analyzing that chart.

Tarefa 7 – Quantas ligações de ataque foram registadas no artigo "Que pensões podemos esperar?"?

Time: 20 s

Process: Search document "Que pensões podemos esperar?". Click on it in list of documents to open its tab. Go to the "Global" tab. The there's two options:

- Analyze "Links By Topic" chart. See the number in the red bar corresponding to the attack links of the selected article.
- Analyze "Support/Attack Per Document" chart, hover over the red bar and see the number of attack links.

Answer: 11

Objective: Understand if user prefers the "Links By Topic" chart or the "Support/Attack Per Document" chart to answer this question. Or if none is intuitive enough.

Tarefa 8 – Aponte o parágrafo que tem mais argumentos em comum registados pelos anotadores no artigo "O Brasil espera Portugal para a Olimpíada 2016"

Time: 30 s

Process: Search document "O Brasil espera Portugal para a Olimpíada 2016". Click on it in list of documents to open its tab. Analyze text and highlight colors. A more intense color (red) corresponds to a higher number of arguments found in that part of the text. **OR** Analyze the "Link's Distribution" chart and see in which paragraph (analyzing the axis) there are more links.

Answer: Third paragraph: "A cinco meses dos jogos olímpicos, podemos [...], sendo o restante destinado a melhorias que ficarão para a cidade depois do evento"

Objective: See if the highlight colors are intuitive to show arguments in common or if the "Link's Distribution" is better or if the user doesn't understand how to perform this task because this functionality is not perceptible.

Tarefa 9 – Que tipo de ADUs e relações existem no último mapa de argumento do artigo "Coragem: é a nossa vez"?

Time: 35 s

Process: Search document "Coragem: é a nossa vez". Click on it in list of documents to open its tab. Select the last blue rectangle in the "Link's Distribution" chart corresponding to the last argument map. Analyze map, using the chart in the "Global" tab as a label.

Answer: Two ADUs of type "Valor" and one link of support between them.

Objective: See if the user can interpret argument maps, identifying their components.

Tarefa 10 – Qual é o tipo de ADU da conclusão do primeiro mapa de argumento registado para o artigo "O tempo da igualdade é já!"?

Time: 35 s

Process: Search document "O tempo da igualdade é já!". Click on it in list of documents to open its tab. Analyze "Link's Distribution" chart and select the first blue rectangle corresponding to the first argument map. Analyze the conclusion ADU and refer to the chart in the "Global" tab as a label to answer.

Answer: ADU of type "Valor(+)"

Objective: See if the user can interpret an argument map, identifying its conclusion, even in a bigger map.

Part 2.3 – End of the Dashboard Testing

After the ten tasks, the user will have to fill a form (SUS survey) about the platform that they just tested.

Part 3 – ScrollyTeller Testing

Part 3.1 – Demo and Exploration

Before starting the tests, the researcher explains the structure of the platform to the tester, showing briefly how it works.

Afterwards, it is given some minutes for the tester to read the story entirely without any time limit, since everyone has their reading rhythm.

Part 3.2 – Tasks

Protocol

The tester will be asked to perform eight tasks for this platform. Each question will be said out loud and clarified if any questions are asked. The questions are showed in paper, highlighted with a sticker, and in a word document in the computer, so the tester can easily access it while doing the task. The tester can also ask the researcher to repeat the task as many times as they want while doing the task.

The tester must answer the question out loud so the researcher know they have finished the task.

The order of the questions will be random (determined by a Latin Square) for all testers, except question 8 that will always be the last and will not be timed. Question 8 helps to know if the tester understood the story that is being conveyed by the platform, checking if they need to see in the platform or they got it by reading the story and doing the tasks.

All questions begin with the initial page. It is possible to refresh if necessary. Or just scroll up.

The researcher must register if the tester was successful doing the task, as well as the time the tester took to finish the task.

After every task, the tester must answer a form (NASA-TLX survey) regarding the effort, satisfaction and frustration they felt performing the task.

Tasks Description

Tarefa 1 – Com quem é que o PSD costumava votar?

Time: 25 s

Process: Go to the bar chart with buttons and select the one regarding the party “PSD” (in orange). The biggest bar corresponds to CDS-PP. **OR** read the text on the left that explains the chart.

Answer: CDS-PP

Tarefa 2 – Quantas iniciativas foram rejeitadas quando apenas o PS votou contra, quando este tinha maioria absoluta e não houve abstenções?

Time: 40 s

Process: Go to the bar chart regarding the rejected initiatives by PS and scroll down to the no abstentions graph. Analyze the bar corresponding to the legislature X, where PS had absolute majority.

Answer: 17

Tarefa 3 – Qual foi a percentagem de rejeição de iniciativas do BE na legislatura X?

Time: 40 s

Process: Go to the line chart and analyze the line that belongs to the party “BE” (in lilac). See the dot that corresponds to the legislature X. Scroll down to see the percentage.

Answer: 95.7%

Tarefa 4 – Qual foi a percentagem total de rejeição de iniciativas na legislatura X?

Time: 50 s

Process: Scroll down to the big numbers. Analyze which one refers to the legislature X (last one)

Answer: 68.1%

Tarefa 5 – Diga-me o nome de uma iniciativa rejeitada na legislatura X.

Time: 16 s

Process: Go to the first chart and hover over one of the blue boxes corresponding to the legislature X (separated by a line), and which is at the lowest level, which corresponds to the rejected initiatives. A tooltip will appear with the name of the initiative.

Answer: depends on the selected initiative.

Tarefa 6 – Que partido teve as suas iniciativas mais rejeitadas ao longo das três legislaturas?

Time: 20 s

Process: Go to the first stacked bar chart and analyze the bars corresponding to the parties. See which one has the higher percentage of rejected initiatives (red part), which is the first bar.

Answer: PCP

Tarefa 7 – Que partidos votaram contra a iniciativa Institui o Tributo Solidário?

Time: 30 s

Process: Go to the exploratory area and search the initiative “Institui o Tributo Solidário”. Click on the highlighted node on the graph and analyze the text on the right.

Answer: [’PS’, ’BE’, ’PCP’, ’PEV’]

Tarefa 8 – A legislatura X teve mais iniciativas rejeitadas que as restantes legislaturas?

Time: As described before, time will not be measured.

Process: If needed, analyze the big numbers in the end.

Answer: Sim (10% more than the rest of the legislatures)

Part 3.3 – End of the Dashboard Testing

After the eight tasks, the user will have to fill a form (SUS survey) about the platform that they just tested.

Part 4 – End of Testing

After testing both platforms, the form they have been answering throughout the test presents a space for them to add suggestions and opinions. However, everything that is said during the test is annotated by the researcher.

C

Testing Protocol – Experts

Testing Protocol – Experts

Required Material

For this test, the conductor must use the same computer with connection to the internet, with a microphone and camera, and access to the following links:

- Dashboard: https://web.ist.utl.pt/ist189407/DARGMINTS/
- ScrollyTeller: https://scrollyteller.herokuapp.com/
- Form: https://forms.gle/HUbvHX2W6q8wFd237

The conductor must also carry material to collect data gathered from the test, which includes pen and paper.

Besides this, the tester must also have access to a computer with connection to the internet and a microphone.

Testers

For this testing phase, it will be selected at least 5 experts on the domain.

Beginning of the test

The tests are online, through Zoom, since it has the ability to pass the control through screen share. This allows the tests to be performed using the same system all the time.

A form is presented to the tester at the beginning of the test, with the following introduction:

> "Bem-vindo(a)!
>
> O meu nome é Ana Silva e estou a completar o mestrado em Engenharia Informática e de Computadores no Instituto Superior Técnico.
>
> Neste momento, estou na fase de testes de uma framework desenvolvida no contexto da minha dissertação, em conjunto com o projeto DARGMINTS.
>
> Essa framework chama-se Visual Argument e visa tornar mais fácil analisar texto argumentativo, usando componentes visuais interativas e intuitivas.
>
> Para mostrar a sua versatilidade, criei duas plataformas, o Dashboard e o ScrollyTeller, com diferentes estruturas mas ambas usando a mesma framework como base.

Para as testar, gostaria que fizesse parte do processo de testagem, ao dedicar algum do seu tempo e atenção. O teste consiste em realizar algumas tarefas para cada plataforma, tendo algum tempo para explorar as plataformas e fazer perguntas antes.

Este teste será realizado de forma online e durará cerca de 1 hora, sendo possível desistir a qualquer momento sem qualquer penalização.

O teste será anónimo e todos os dados colecionados ao longo do teste serão apenas para efeitos estatísticos. Obrigada desde já pela sua atenção e contribuição para a avaliação deste projeto."

After they read the introduction they can consent to the read information when they ready and begin the tests.

Part 1 - Initial Survey

Before starting the tests themselves, the tester will have to fill some demographic questions, relating to their age, gender, visual difficulties and level of schooling.

Afterwards, the tester will be asked to analyze five charts: a bar chart, a pie chart, a line chart, a graph and a box plot. This will help to understand the level of knowledge of the testers regarding the analysis of charts.

Part 2 – Dashboard Testing

Part 2.1 – Demo and Exploration

After filling the initial survey, the researcher will perform a short demonstration explaining the key elements of the platform's structure.

Afterwards, the tester will have exactly 5 minutes to freely explore the platform. They are encouraged to say out loud what they are thinking and feeling.

Part 2.2 – Tasks

Protocol

The tester will be asked to perform six tasks for this platform. Each question will be said out loud and clarified if any questions are asked. The questions are in a word document in the computer, so the tester can easily access it while doing the task. The tester can also ask the researcher to repeat the task as many times as they want while doing the task.

The tester must answer the question out loud so the researcher know they have finished the task.

The order of the questions will be random (determined by a Latin Square) for all testers.

All questions begin with no documents and no filters selected.

The researcher must register if the tester was successful doing the task, as well as the time the tester took to finish the task.

The tester is encouraged to follow the method “think-out-loud”, where they say out loud what they are thinking.

After every task, the tester must answer a form (NASA-TLX survey) regarding the effort, satisfaction and frustration they felt performing the task.

Tasks Description

The following tasks are described in Portuguese.

Tarefa 1 – Quais os nomes dos documentos com o tópico Sociedade que têm uma conexão mais forte, ou seja, que têm mais palavras chave em comum?

Time: 35 s

Process: Select the topic “Sociedade”. Identify the strongest line in graph. Use slider to help the analysis. Hover on the nodes connected to that line and say their names out loud.

Answer: “Quo vadis obesidade” and “A obesidade como catástrofe ambiental dos tempos modernos”

Objective: See if the user can use the filters along with the slider of the graph.

Tarefa 2 – Quantos ADUs do tipo Facto registou o anotador 2 no artigo de opinião “Depois de Salvador Sobral”?

Time: 25 s

Process: Search document “Depois de Salvador Sobral”. Click on it in list of documents to open its tab. Select annotator 2 and go to the “Global” tab. Analyze “Types of ADUs” chart and observe bar that correspond to the “Facto” ADU. See the number present in the bar.

Answer: 3

Objective: See if user can interpret the “Types of ADUs” chart”, understanding that it varies with the selection of the annotator.

Tarefa 3 – Entre o artigo mais antigo sobre Desporto e o mais recente, qual deles teve mais

ligações de ataque? O mais antigo ou o mais recente?

Time: 45 s

Process: Select the topic “Desporto”. Select the oldest and newest articles in the list of documents, corresponding to the first (“Pouco pão e muito circo, morte e bocejo”) and last (“Jogador residente, o aplauso de uma raridade”) of the filtered list. Go to the “Global” tab and analyse the “Support/Attack Per Document” chart. Observe the bars corresponding to the attack links and see which one is bigger. To see the numbers, it is possible to use the tooltip by hovering on the bars.

Answer: “Pouco pão e muito circo, morte e bocejo” (oldest)

Objective: Understand if the “Support/Attack Per Document” chart is intuitive and if the user understands they can answer the question analyzing that chart.

Tarefa 4 – Aponte o parágrafo que tem mais argumentos em comum registados pelos anotadores no artigo “O Brasil espera Portugal para a Olimpíada 2016”

Time: 30 s

Process: Search document “O Brasil espera Portugal para a Olimpíada 2016”. Click on it in list of documents to open its tab. Analyze text and highlight colors. A more intense color (red) corresponds to a higher number of arguments found in that part of the text. **OR** Analyze the “Link’s Distribution” chart and see in which paragraph (analyzing the axis) there are more links.

Answer: Third paragraph: “A cinco meses dos jogos olímpicos, podemos [. . .], sendo o restante destinado a melhorias que ficarão para a cidade depois do evento”

Objective: See if the highlight colors are intuitive to show arguments in common or if the “Link’s Distribution” is better or if the user doesn’t understand how to perform this task because this functionality is not perceptible.

Tarefa 5 – Que tipo de ADUs e relações existem no último mapa de argumento do artigo “Coragem: é a nossa vez”?

Time: 35 s

Process: Search document “Coragem: é a nossa vez”. Click on it in list of documents to open its tab. Select the last blue rectangle in the “Link’s Distribution” chart corresponding to the last argument map. Analyze map, using the chart in the “Global” tab as a label.

Answer: Two ADUs of type “Valor” and one link of support between them.

Objective: See if the user can interpret argument maps, identifying their components.

Part 2.3 – End of the Dashboard Testing

After the five tasks, the user will have to fill a form (SUS survey) about the platform that they just tested.

Part 3 – ScrollyTeller Testing

Part 3.1 – Demo and Exploration

Before starting the tests, the researcher explains the structure of the platform to the tester, showing briefly how it works.

Afterwards, it is given some minutes for the tester to read the story entirely without any time limit, since everyone has their reading rhythm.

Part 3.2 – Tasks

Protocol

The tester will be asked to perform eight tasks for this platform. Each question will be said out loud and clarified if any questions are asked. The questions are showed in paper, highlighted with a sticker, and in a word document in the computer, so the tester can easily access it while doing the task. The tester can also ask the researcher to repeat the task as many times as they want while doing the task.

The tester must answer the question out loud so the researcher know they have finished the task.

The order of the questions will be random (determined by a Latin Square) for all testers, except question 8 that will always be the last and will not be timed. Question 8 helps to know if the tester understood the story that is being conveyed by the platform, checking if they need to see in the platform or they got it by reading the story and doing the tasks.

All questions begin with the initial page. It is possible to refresh if necessary. Or just scroll up.

The researcher must register if the tester was successful doing the task, as well as the time the tester took to finish the task.

The tester is encouraged to follow the method “think-out-loud”, where they say out loud what they are thinking.

After every task, the tester must answer a form (NASA-TLX survey) regarding the effort, satisfaction and frustration they felt performing the task.

Tasks Description

Tarefa 1 – Qual foi a percentagem de rejeição de iniciativas do BE na legislatura X?

Time: 40 s

Process: Go to the line chart and analyze the line that belongs to the party “BE” (in lilac). See the dot that corresponds to the legislature X. Scroll down to see the percentage.

Answer: 95.7%

Tarefa 2 – Qual foi a percentagem total de rejeição de iniciativas na legislatura X?

Time: 50 s

Process: Scroll down to the big numbers. Analyze which one refers to the legislature X (last one)

Answer: 68.1%

Tarefa 3 – Diga-me o nome de uma iniciativa rejeitada na legislatura X.

Time: 16 s

Process: Go to the first chart and hover over one of the blue boxes corresponding to the legislature X (separated by a line), and which is at the lowest level, which corresponds to the rejected initiatives. A tooltip will appear with the name of the initiative.

Answer: depends on the selected initiative.

Tarefa 4 – Que partido teve as suas iniciativas mais rejeitadas ao longo das três legislaturas?

Time: 20 s

Process: Go to the first stacked bar chart and analyze the bars corresponding to the parties. See which one has the higher percentage of rejected initiatives (red part), which is the first bar.

Answer: PCP

Tarefa 5 – Que partidos votaram contra a iniciativa Institui o Tributo Solidário?

Time: 30 s

Process: Go to the exploratory area and search the initiative “Institui o Tributo Solidário”. Click on the highlighted node on the graph and analyze the text on the right.

Answer: [’PS’, ’BE’, ’PCP’, ’PEV’]

Tarefa 6 – A legislatura X teve mais iniciativas rejeitadas que as restantes legislaturas?

Time: As described before, time will not be measured.

Process: If needed, analyze the big numbers in the end.

Answer: Sim (10% more than the rest of the legislatures)

Part 3.3 – End of the Dashboard Testing

After the six tasks, the user will have to fill a form (SUS survey) about the platform that they just tested.

Part 4 – End of Testing

After testing both platforms, the form they have been answering throughout the test presents a space for them to add suggestions and opinions. However, everything that is said during the test is annotated by the researcher.