

# How good are stories told by Twitter? An NLP Approach

**Mafalda Rodrigues Castro**

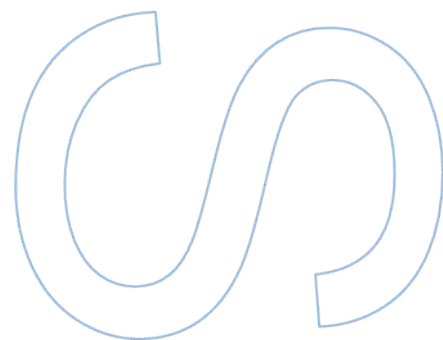
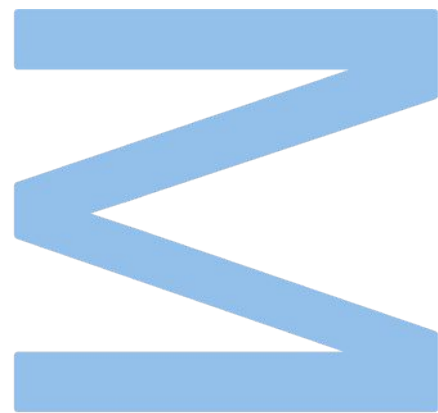
Mestrado em Ciência de Computadores  
Departamento de Ciência de Computadores  
2022

**Orientador**

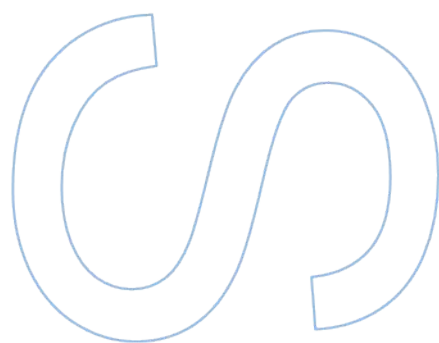
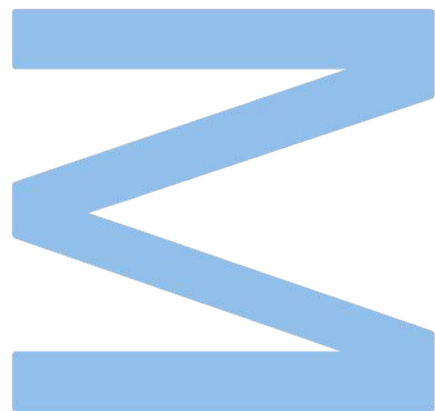
Alípio Jorge, Professor Associado,  
Faculdade de Ciências da Universidade do Porto

**Coorientador**

Ricardo Campos, Professor Adjunto,  
Instituto Politécnico de Tomar









# Sworn Statement

I, Mafalda Rodrigues Castro, enrolled in the Master's Degree in Computer Science at the Faculty of Sciences of the University of Porto hereby declare, in accordance with the provisions of paragraph a) of Article 14 of the Code of Ethical Conduct of the University of Porto, that the content of this dissertation reflects perspectives, research work and my own interpretations at the time of its submission.

By submitting this dissertation, I also declare that it contains the results of my own research work and contributions that have not been previously submitted to this or any other institution.

I further declare that all references to other authors fully comply with the rules of attribution and are referenced in the text by citation and identified in the bibliographic references section. This dissertation does not include any content whose reproduction is protected by copyright laws.

I am aware that the practice of plagiarism and self-plagiarism constitute a form of academic offense.

Mafalda Rodrigues Castro

30/09/2022



# Abstract

The rise of social media has brought a great transformation to the way news are discovered and shared. Unlike traditional news sources, social media allows anyone to cover a story. Therefore, sometimes an event is already discussed by people before a journalist turns it into a news article. Twitter is a particularly appealing social network for discussing events, since its posts are very compact and, therefore, contain colloquial language and abbreviations. However, its large volume of tweets also makes it impossible for a user to keep up with an event.

In this thesis, we present a framework for extracting narratives from *tweets* in real time. Since the field of narrative extraction is still relatively new and its models are computationally expensive, we first have to collect the tweets, then select the most relevant ones and only then extract their narrative elements. With this information at hand, we then present the narrative in an appealing format, be it textual or visual. For this reason, we started by researching existing approaches for summarizing a tweet stream. Our focus was the Portuguese language, so we researched datasets linking Portuguese tweets to news, but couldn't find any that met our criteria.

To circumvent this problem, we decided to create our own Portuguese dataset. To reach our aims, we started by collecting a set of events and news articles, and then retrieved tweets related to them. This gave origin to our dataset, Tweet2Event-PT, which was made available to the public. After performing text cleaning, we explored different methods to filter out irrelevant tweets and concluded that the best one was the Okapi BM25 function.

We then developed a framework, TweetStream2Story, that allows a user to extract and visualize the narrative of a specified topic. This pipeline is built on top of the existing Tweet2Story framework, which extracts narrative elements from a set of tweets. However, contrary to Tweet2Story, our framework automatically collects and filters *tweets* in real time for a specified topic and was tested for the Portuguese language.

To evaluate this framework, we conducted two surveys. The first one aims to assess the usability and user preferences regarding visualization of the narrative. The second survey looks more closely at the narrative elements extracted for three case studies. From there, we concluded that our framework is able to extract relevant information that can complement a news article.

**Keywords:** Narrative extraction, Natural Language Processing, Twitter, Tweet Summarization





# Resumo

O crescimento das redes sociais trouxe uma grande transformação na maneira como as notícias são descobertas e partilhadas. Ao contrário de meios de comunicação tradicionais, as redes sociais permitem que qualquer pessoa cubra uma história. Assim, por vezes, um evento já é discutido por um grupo de pessoas antes de um jornalista o converter numa notícia. O Twitter é uma rede social particularmente apelativa para a discussão de eventos, visto que as suas publicações são muito compactas, e portanto, contêm linguagem coloquial e abreviações. No entanto, o volume de *tweets* publicados dificulta a capacidade de um utilizador acompanhar um evento.

Nesta tese, apresentamos uma plataforma para a extração de narrativas a partir de publicações no Twitter em tempo real. Visto que o campo de extração de narrativas é relativamente recente e os seus modelos têm elevada complexidade computacional, é primeiro necessário recolher os *tweets*, depois selecionar os mais relevantes, para finalmente proceder à interpretação e extração dos seus elementos narrativos. Com esta informação, apresentamos então a narrativa num formato apelativo, quer textual quer visualmente. Por este motivo, o nosso primeiro passo foi pesquisar métodos existentes para a sumarização de um conjunto de *tweets*. Também nos quisemos focar na língua Portuguesa, pelo que pesquisámos *datasets* portugueses que ligassem *tweets* a notícias, mas não encontramos nenhum que satisfizesse os nossos critérios.

Para contornar este problema, decidimos criar o nosso próprio dataset em português. Para atingir este objetivo, recolhemos um conjunto de eventos e notícias, assim como um conjunto de *tweets* relacionados, criando o nosso próprio dataset português para esta tarefa. Depois de realizar o processo de limpeza de texto, explorámos diferentes métodos para remover *tweets* irrelevantes, e concluímos que o melhor era a função Okapi BM25.

Em seguida, desenvolvemos uma plataforma, TweetStream2Story, que permite a um utilizador visualizar a narrativa de um tópico por si especificado. Esta plataforma recolhe automaticamente *tweets* relacionados a esse tópico em tempo real, extrai os seus elementos da narrativa e gera uma cronologia onde se pode ver a evolução do evento ao longo do tempo, a partir de uma sequência de grafos de conhecimento e os seus respetivos *tweets*.

Para avaliar esta plataforma, realizámos duas sondagens. A primeira tem como objetivo avaliar a acessibilidade e preferências do utilizador relativamente à visualização dos resultados. A segunda analisa de maneira mais profunda os elementos narrativos extraídos para três casos de estudo. A partir destes resultados, concluímos que esta ferramenta é capaz de extrair informação

relevante capaz de complementar uma notícia.

**Palavras-chave:** Extração de narrativas, Processamento de Linguagem Natural, Twitter, Sumarização de Tweets

# Acknowledgements

This thesis marks the end of five incredible years, which would have been very different without the people who supported me since the beginning, and the people I met throughout them.

Firstly, I'd like to thank my family, who have always cared for and given me all the support and love throughout this journey. I also want to thank Joca, who has given me encouragement and reassurance whenever I needed, and stood by me all the time. I also want to thank all the friends I made while studying here, and the ones I've known long before, who I can still count on to this day.

Next, I want to thank both my supervisors: Alípio Jorge and Ricardo Campos. Thank you for letting me be a part of an amazing project and providing me guidance throughout the entire year. Your advice and insights were incredibly helpful and vital for this work, and I feel honored to have been a part of this project.

Two years ago, I joined INESC TEC and got to take part in different research projects, allowing me to grow personally, academically and professionally. I want to thank my research supervisor Lino Oliveira for providing me this opportunity, and to my colleagues, who made it even more special.

Finally, I want to thank everyone that answered our surveys, your input was very helpful.

Although I am proud of my hard work and dedication to this point, I know they wouldn't have been enough if not for the people in my life.

Thank you all,  
Mafalda

**To Mom, Dad, Miguelito and Joca**

# Contents

<b>Abstract</b>	<b>vii</b>
<b>Resumo</b>	<b>ix</b>
<b>Acknowledgements</b>	<b>xi</b>
<b>Contents</b>	<b>xv</b>
<b>List of Tables</b>	<b>xvii</b>
<b>List of Figures</b>	<b>xx</b>
<b>Acronyms</b>	<b>xxi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Context . . . . .	1
1.2 Motivations and Goals . . . . .	2
1.3 Methodology . . . . .	2
1.4 Contributions . . . . .	3
1.5 Outline . . . . .	4
<b>2 Background Work</b>	<b>5</b>
2.1 Selected NLP Tasks and Concepts . . . . .	5
2.1.1 Text Summarization . . . . .	5
2.1.2 Keyword Extraction . . . . .	6

2.1.3	Text Similarity . . . . .	7
2.2	Tweet2Story . . . . .	10
2.3	Summary . . . . .	11
<b>3</b>	<b>Related Work</b>	<b>13</b>
3.1	Discussion . . . . .	13
3.1.1	What sort of work exists related to tweet summarization? . . . . .	13
3.1.2	What methods and metrics are used to determine the relevance of a tweet for a given event? . . . . .	15
3.1.3	Are there existing Portuguese datasets linking <i>tweets</i> to news articles? . . . . .	15
3.2	Summary . . . . .	16
<b>4</b>	<b>Tweet2Event-PT: Linking Events and Tweets in the Portuguese Language</b>	<b>17</b>
4.1	Event Retrieval . . . . .	17
4.2	Tweet Retrieval . . . . .	19
4.3	Tweet Cleaning . . . . .	21
4.4	Tweet Relevance . . . . .	22
4.4.1	Experiments on the Signal Dataset . . . . .	22
4.4.2	Experiments on Tweet2Event-PT . . . . .	24
4.5	Summary . . . . .	26
<b>5</b>	<b>TweetStream2Story: Narrative Extraction from Tweets Collected in Real-Time</b>	<b>29</b>
5.1	Use Case . . . . .	29
5.2	Architecture of the Solution . . . . .	30
5.3	Elasticsearch . . . . .	31
5.4	Flask Web Server . . . . .	32
5.4.1	REST API . . . . .	33
5.4.2	Topic Class . . . . .	33
5.5	Webpage . . . . .	36

5.5.1	Token Generation . . . . .	36
5.5.2	Topic Input . . . . .	36
5.5.3	List of topics . . . . .	37
5.5.4	Topic Visualization . . . . .	38
5.6	Summary . . . . .	40
<b>6</b>	<b>Results and Discussion</b>	<b>41</b>
6.1	Usability and Visualization Survey . . . . .	41
6.1.1	Discussion . . . . .	42
6.2	Narrative Assessment Survey . . . . .	43
6.2.1	Case Study 1: Fire in Castro Marim . . . . .	45
6.2.2	Case Study 2: Soccer Match: Benfica vs Belenenses . . . . .	46
6.2.3	Case Study 3: Shooting in Denmark . . . . .	48
6.2.4	Discussion . . . . .	49
6.3	Summary . . . . .	49
<b>7</b>	<b>Conclusions</b>	<b>51</b>
7.1	Limitations . . . . .	52
7.2	Future Work . . . . .	53
<b>A</b>	<b>Results of Usability Survey</b>	<b>55</b>
A.1	Section 1 . . . . .	55
A.2	Section 2 . . . . .	58
<b>B</b>	<b>Results of Narrative Evaluation Survey</b>	<b>61</b>
B.1	Section 1 - Fire in Castro Marim . . . . .	61
B.2	Section 2 - Soccer Match: Benfica vs Belenenses . . . . .	63
B.3	Section 3 - Shooting in Denmark Mall . . . . .	66
	<b>Bibliography</b>	<b>69</b>





# List of Tables

3.1	Types of Tweet Summarization for different papers . . . . .	14
3.2	Types of Portuguese datasets with <i>tweets</i> . . . . .	16
4.1	Number of tweets per event before pre-processing . . . . .	20
4.2	Number of tweets per event after pre-processing . . . . .	21
4.3	Number of relevant tweets in the dataset . . . . .	27
5.1	TweetStream2Story Representational State Transfer (REST) Application Programming Interface (API) Endpoints . . . . .	33
5.2	Attributes of class Topic . . . . .	34
6.1	Events used for evaluation through survey . . . . .	43
6.2	Overall user scores . . . . .	44
6.3	Overall user scores for "Who" and "What" elements . . . . .	45
6.4	Overall user scores for "When" and "Where" elements . . . . .	45



# List of Figures

2.1	BertSum architecture compared to original [5]	6
2.2	Word2Vec Methods [21]	9
2.3	Sentence similarity using Sentence-BERT [22]	9
2.4	Annotation Scheme used by Tweet2Story [25]	10
4.1	Wikipedia "2021 in Portugal" page event example	18
4.2	Wikipedia "2021 in Portugal" page source-code notation example	19
4.3	Query used to retrieve tweets for a given topic	20
4.4	Distribution of <i>tweets</i> by topic on Signal dataset	22
4.5	Distribution of tweets by relevance	23
4.6	Normalized Discounted Cumulative Gain (NDCG) results	24
4.7	Precision@K of different methods on the Signal dataset	25
4.8	F1@K of different methods on the Signal dataset	25
4.9	Precision-Recall Curve of different methods on the Signal dataset	25
4.10	Precision, recall and percentage of relevant tweets for different BM25 scores thresholds in the selected sample	27
5.1	Use Case Diagram	30
5.2	TweetStream2Story Architecture Overview	31
5.3	Directory structure of flask application	32
5.4	Query input for topic	37
5.5	Options for narrative extraction	37

5.6	Tweet retrieval mode . . . . .	37
5.7	Topics list . . . . .	38
5.8	Timeline interface . . . . .	39
5.9	Knowledge graph with different colors to represent new actors . . . . .	40
6.1	Most used media for obtaining news by age group . . . . .	42
A.1	Age . . . . .	55
A.2	Academic Qualifications . . . . .	56
A.3	Ranking of preferred media for obtaining news . . . . .	56
A.4	Twitter Use . . . . .	57
A.5	Frequency of using Twitter to keep up with events in real-time . . . . .	57
B.1	Knowledge Graph for the event "Fire in Castro Marim" . . . . .	61
B.2	Interesting actors/relations not present in the summary . . . . .	62
B.3	Relevance of actors/relations new actors in a global time window . . . . .	62
B.4	Knowledge Graph for the event "Soccer Match: Benfica vs Belenenses" . . . . .	63
B.5	Interesting actors/relations not present in the summary . . . . .	64
B.6	Relevance of actors/relations new actors in a global time window . . . . .	65
B.7	Knowledge Graph for the event "Shooting in Denmark Mall" . . . . .	66
B.8	Interesting actors/relations not present in the summary . . . . .	67
B.9	Relevance of actors/relations new actors in a global time window . . . . .	67

# Acronyms

<b>AI</b>	Artificial Intelligence	<b>NDCG</b>	Normalized Discounted Cumulative Gain
<b>API</b>	Application Programming Interface	<b>NER</b>	Named Entity Recognition
<b>BA</b>	Basic Authentication	<b>NLP</b>	Natural Language Processing
<b>BERT</b>	Bidirectional Encoder Representations from Transformers	<b>REST</b>	Representational State Transfer
<b>CSV</b>	Comma-Separated Values	<b>TF-IDF</b>	Term Frequency–Inverse Document Frequency
<b>CBOW</b>	Common Bag of Words	<b>TREC</b>	Text REtrieval Conference
<b>DRS</b>	Discourse Representation Structures	<b>URL</b>	Uniform Resource Locator
<b>HTTP</b>	Hypertext Transfer Protocol	<b>UUID</b>	Universally Unique Identifier
<b>ILP</b>	Integer Linear Programming	<b>YAKE</b>	Yet Another Keyword Extractor
<b>IR</b>	Information Retrieval		
<b>JSON</b>	JavaScript Object Notation		



# Chapter 1

## Introduction

### 1.1 Context

The growth of social media has brought a change to the way news are discovered and shared, as anyone can cover an event on a social network. Often, events are discussed in real-time on social media by the common user, before being turned into news. Artificial Intelligence, particularly Natural Language Processing (NLP) models, can be useful to unmask the narrative behind a set of small posts. This is useful for generating news and to have an integrated view of twitter bursts on current topics.

Twitter is a social media platform founded in 2006 whose popularity continues to rise to this day [1], and widely used by journalists. According to a survey by MuckRack in 2022 about the State of Journalism [2], 77% of the participants listed Twitter when asked their most valuable social networks, as journalists. When enquired about the sources they go first for news, Twitter is the second most voted source, preceded by online newspapers and magazines. Another survey, conducted by the Pew Research Center [3], also reported that Twitter was the social network most used by journalists for their jobs, in the United States, supporting this claim.

We believe that a tool for visualizing the narrative behind a set of *tweets* can provide new material for journalists to include in their stories. However, its potential stretches beyond the field of journalism. It can be used, for example, by authorities, to monitor natural disasters, but can also be of interest to any user who wishes to stay up-to-date on a certain topic or ongoing event.

In the remainder of this chapter, we define the objectives and motivations of this work, as well as the methodology used and our contributions. Finally, we provide the general structure of this thesis with a brief description of each chapter.

## 1.2 Motivations and Goals

Social media is a powerful tool that can provide great insights into a variety of topics. Using Twitter posts as a source for extracting narratives may bring us information different than a news article does, from the people who experience an event first-hand.

The main objective of this work is to extend the Tweet2Story framework to allow the real-time automatic extraction of narratives from a set of tweets about the same event. However, the dimension and colloquial language of Twitter data poses an interesting challenge for those in need of having readily and quality data. Since thousands of tweets are posted every second, it is crucial to clean them and select only the most relevant ones as extracting the narrative from all the retrieved tweets is simply not feasible. In order to achieve this, we started by researching existing approaches for tweet summarization and ended up by proposing the adoption of the BM25 algorithm to rank tweets by their relevance and selecting the top *tweets* as the summary.

Another goal of this project is to extend the Tweet2Story framework to the Portuguese language. To do this, we set the sub-goal of finding or developing a dataset that links tweets to news articles in Portuguese. Finally, we intend to create a demo where any user can visualize the evolution of narrative of a topic of their choice over time.

## 1.3 Methodology

The work of this thesis was divided into four parts. First, we started by conducting a research on similar work to ours, more specifically in tweet summarization and work linking tweets to news articles in the Portuguese language.

Then, since we couldn't find any Portuguese datasets for our task, we conducted some exploratory work as a means to build our own Portuguese dataset, Tweet2Event-PT. We explored ways to obtain news articles or summaries of events collected from the Wikipedia page "2021 in Portugal", which lists events that occurred in Portugal with references to news articles, sorted in chronological order.

Next, we collected related *tweets* related to these events and performed some text cleaning tasks. After this step, since our dataset wasn't annotated with the relevance of tweets, we proceeded to test different methods to determine this score in another dataset. Then, we applied the most effective method, Okapi BM25, on our dataset, obtaining a collection of more relevant tweets.

After finishing our dataset, we extended the existing Tweet2Story framework by allowing the collection of tweets in real time summarization and visualization of the narrative of an event through a timeline. One such process involves conducting five steps: tweet retrieval, tweet cleaning, tweet summarization, narrative extraction, and narrative visualization, as depicted below:



1. **Tweet retrieval** - Collecting tweets in real-time using the Twitter Stream Application Programming Interface (API);
2. **Tweet cleaning** - Perform text pre-processing tasks to work with clean sentences and improve the quality of the results;
3. **Tweet summarization** - Select the most relevant tweets to be used in the following steps;
4. **Narrative extraction** - Extract entities, dates and semantic relations from the text into an annotation scheme;
5. **Narrative visualization** - Convert the annotation into a representation that's easier to understand, such as a Knowledge Graph.

To evaluate the results of this framework, we conducted two surveys. In the first one, we evaluate the usability of this framework and the user preferences on different features of TweetStream2Story. In the second one, we perform a deeper analysis of three different case studies and assess the user understanding of the knowledge graphs our framework produced.

## 1.4 Contributions

Throughout the course of this project, we were able to provide various contributions that can be valuable to the researchers. Our main contributions are as follows:

1. **Literature Review of tweet summarization approaches and datasets linking Portuguese news articles to tweets:** We researched existing work related to tweet summarization, and compared them, taking into account their type of summarization and whether they are real-time approaches. We also compared different datasets linking Portuguese *tweets* to news articles, to determine if there was any dataset suitable for our task.
2. **Portuguese dataset linking tweets to news articles:** We created a dataset, **Tweet2Event-PT**, linking tweets to events that happened in Portugal in 2021. This dataset has around 6500 tweets related to 12 events, and can be useful to expand the current state of Portuguese datasets in NLP tasks. This dataset, as well as the scripts we developed to extract these events and tweets, is available on GitHub<sup>1</sup>.
3. **TweetStream2Story framework:** We present TweetStream2Story, a novel framework for the automatic extraction of narratives from Twitter microblogs in real time. This pipeline handles the collection of *tweets* related to a given event, cleans them, summarizes them and extracts their narrative for different time windows.

---

<sup>1</sup><https://github.com/LIAAD/tweet2event-pt>

4. **Online demo to visualize and extract narratives from Twitter:** We present a demo<sup>2</sup> where users can try out the TweetStream2Story framework, by inputting a topic of their choice and visualizing its narrative through time.

## 1.5 Outline

The remainder of this document is divided into the following six chapters:

- **Chapter 2: Background** - Throughout this work, we will mention and make use of various **NLP** tasks. This chapter aims to provide context on these tasks in order to fully understand the following chapters of this thesis.
- **Chapter 3: Related Work** - This chapter begins by exploring existing work regarding summarization of *tweet* streams. We also explore existing Portuguese datasets linking *tweets* to news articles and events.
- **Chapter 4: Tweet2Event-PT: Linking Events and Tweets in the Portuguese Language** - In this chapter, we describe the process of creating and processing a Portuguese dataset linking *tweets* to related events that happened in Portugal in the year of 2021.
- **Chapter 5: TweetStream2Story: Narrative Extraction from Tweets Collected in Real-Time** - Explains our thought process and implementation of a framework for collecting *tweets* in real time and extracting their narrative.
- **Chapter 6: Results and Discussion** - Illustrates our process for evaluating the results of this work. Here, we analyze the results of two surveys conducted to understand the usability of our framework, and evaluate the user perception of our results.
- **Chapter 7: Conclusions** - This chapter recapitulates the main content of this thesis, from the main ideas to its results. Finally, we provide an insight into the limitations and future possibilities for this work.

---

<sup>2</sup><https://tweetstream2story.inesctec.pt>

## Chapter 2

# Background Work

In this chapter we provide insight into the basic concepts required to understand the remainder of this work, in particular concepts related to Natural Language Processing (NLP). We will also provide a brief explanation on the Tweet2Story framework, which acts as a foundation for the work presented here.

### 2.1 Selected NLP Tasks and Concepts

Natural Language Processing is a field of Computer Science concerned with the interaction between computers and human language, through the combination of linguistics and Artificial Intelligence (AI). There are many tasks studied in this field, such as text summarization, text classification, information extraction, machine translation or sentiment analysis, providing a wide range of applications in the real world. Throughout this work, we will mention summarization, keyword extraction and text similarity, so these are the tasks that we'll detail in this section.

#### 2.1.1 Text Summarization

Text summarization is a task in NLP that consists of condensing a large text into a short one, a **summary**, while retaining the most relevant information of the original text. One of the aims of our work is to obtain a summary from a stream of tweets, so this is the most important NLP task of this thesis. In this section, we will approach the two categories of text summarization: **extractive** and **abstractive**.

##### 2.1.1.1 Extractive Summarization

This approach consists of creating a summary using information **directly contained** in the text, that is, by identifying the most important sentences and putting them together. This way, the resulting summary will be grammatically correct (assuming the selected sentences are too).

Some examples of the state-of-the-art models that use an extractive approach are MatchSum [4], BertSumExt [5] and DiscoBERT [6], detailed below.

**MatchSum** [4], for example, formulates the summarization task as a semantic text matching problem, in which a source document and candidate summaries are matched in a semantic space. With this approach, the best summaries are semantically closer to the source document.

Another approach is **BertSumExt** [5], which extends Bidirectional Encoder Representations from Transformers (**BERT**) by inserting symbols to learn sentence representations and using interval segmentation embeddings to distinguish multiple sentences (see Figure 2.1).

As for **DiscoBERT** [6], this method extracts sub-sentential discourse units, instead of sentences, as candidates for extractive selection on a finer granularity. They also use structural discourse graphs in order to capture long-term dependencies throughout the document.

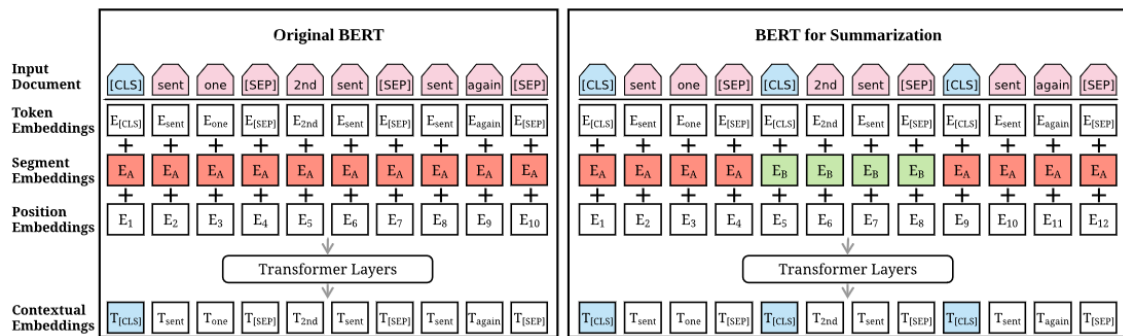


Figure 2.1: BertSum architecture compared to original [5]

### 2.1.1.2 Abstractive Summarization

In contrast to extractive summarization, abstractive models take advantage of deep learning models to **generate new sentences** that may include terms not in the original text.

Currently, one of the state-of-the-art models for abstractive summarization is **SimCLS** [7], a framework for two-stage summarization. In the first stage, it uses BART [8] to generate candidate summaries, and in the second one, the RoBERTa [9] scoring model is used to predict the score of these candidates based on the source document.

Another state-of-the-art approach is **GSum** [10], a model based on neural encoder-decoders, with modifications that allow the model to attend to different guidance signals when generating outputs.

## 2.1.2 Keyword Extraction

Keyword extraction is the task of retrieving the most significant words from a document [11], allowing the identification of its essential themes. This task can be applied in many contexts,

such as finding the most important terms in consumer reviews or detecting trends, and can be used to generate word clouds. In our work, we used this task to extract important words in news articles, that can be joined in a query for retrieving relevant *tweets*.

Keyword extraction can be performed through two different strategies. **Supervised** approaches are based on training their model on **labelled data**, until it detects its underlying patterns. On the other hand, **unsupervised** methods learn **without** using labelled data. As there is a large variety of keyword extraction methods, we will only describe some unsupervised approaches, as our work only makes use of these methods, but additional information can be read in the review conducted by E. Papagiannopoulou et al. [12].

Regarding unsupervised keyword extraction, we can still divide these methods into sub-categories, ranging from **statistic-based** to **graph-based** approaches. Statistic-based approaches can score words based on a variety of statistics, such as word frequency, co-occurrence or word collocation, while graph-based approaches represent a document's text through a graph.

**Term Frequency–Inverse Document Frequency (TF-IDF)** [13], for example, is a weighting statistic that computes the **frequency of each term** and multiplies it by the **inverse of the term's frequency in the complete corpus**. This statistic favors frequent words in the document that are not frequent on the remaining of the corpus. This way, the keywords of the document are the terms with the highest scores.

**Yet Another Keyword Extractor (YAKE)** [14] is (yet) another keyword extractor that makes use of statistical features in its approach, and works with documents of any domain, language and size, as it is **independent of a corpus**. This approach begins by processing the text and extracting candidate terms, and then proceeds to extract a set of statistical features that capture the nature of a term. Next, these features are combined into a single score that determines the importance of a term. After this, it generates candidate keywords and assigns them scores, and finally, it performs a deduplication step to remove similar keywords.

When it comes to **graph-based** keyword extraction, the most popular approach is probably **TextRank** [15], a model analogous to Google's PageRank (1998), that has generated various variations [16]. This model uses sentences in a document as nodes, and the edges are based on the overlap of two sentences, by calculating their words in common.

### 2.1.3 Text Similarity

Text similarity is a very common **NLP** task that aims to determine how close two text documents are from each other [17, 18]. This similarity can be calculated in different ways. For instance, we may want to know the **lexical** similarity, that is, how similar the documents are in on a **word level**. Or we may want to know their **semantic** similarity, that is, how close they are in their **meaning**.

To better understand this, let's take as an example the following sentences:

1. Mother cleaned the sink.
2. Mom washed the basin.

Looking at these two phrases, they suggest a **high semantic similarity** but a **low lexical similarity**, since they have the same meaning but use different words.

### 2.1.3.1 Lexical Similarity

Lexical approaches calculate the similarity through the **overlap of words** between two texts and may, or not, take into account attributes such as the order of the words and their frequency. Some examples of these approaches to text similarity are Jaccard Similarity and **TF-IDF** cosine similarity.

**Jaccard similarity** [19] is a simple metric that can be seen as the number of common words over the total number of words in the two texts, ranging from 0 to one. One shortcoming of this metric is that it doesn't take into account how many times the words appears.

Another approach is **TF-IDF cosine similarity**, where we compute the cosine similarity between the **TF-IDF** vectors of two documents. With this metric, two documents will be considered similar if their terms have similar importance.

### 2.1.3.2 Semantic Similarity

In order for a computer to understand the semantic similarity between two sentences, there must be a **representation of their meaning** such that we can do operations with them, and to achieve this we use **word embedding** models [20]. A **word embedding** is a **vectorial representation** of a word that aims to capture its meaning and context.

One of the most common techniques for constructing word embeddings is **Word2Vec** [21], an algorithm that can learn word associations from a large corpus of text and represent each word as a vector. These embeddings can be obtained in two different ways:

- **Skip Gram** - The input is the target word, and the output is a set of probability distributions, one for each word in the vocabulary;
- **Common Bag of Words (CBOW)** - The input is the context of a word and the output is the target word.

If we want to capture the meaning of a sentence instead of just a word, we use **sentence embeddings**, a technique to obtain a vectorial representation of an entire sentence.

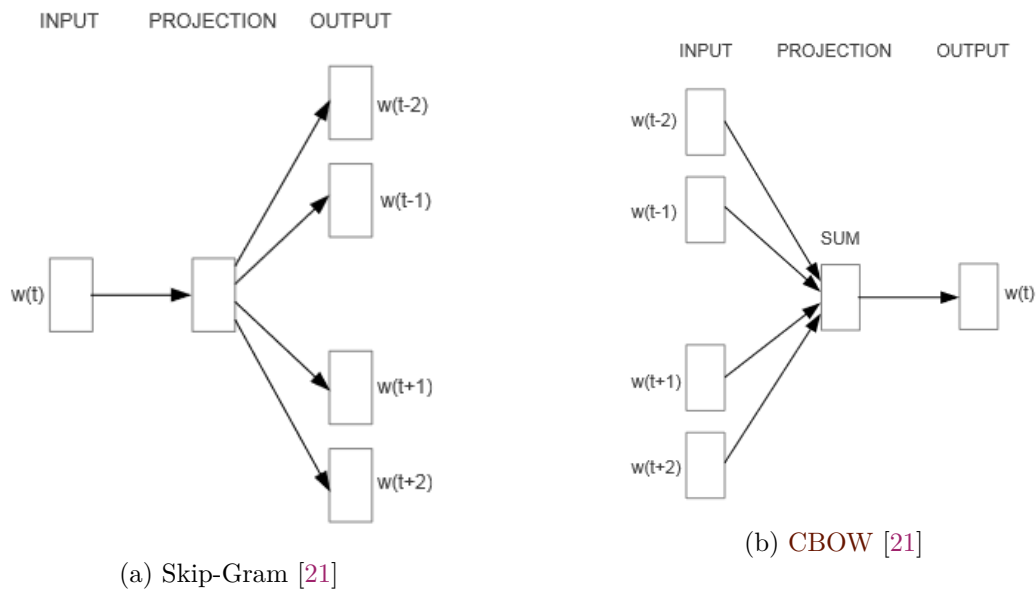


Figure 2.2: Word2Vec Methods [21]

The **SentenceBERT** model makes use of a Siamese network architecture, in which each sentence given as input passes through a BERT model and a pooling layer to generate their embeddings, as shown in 2.3.

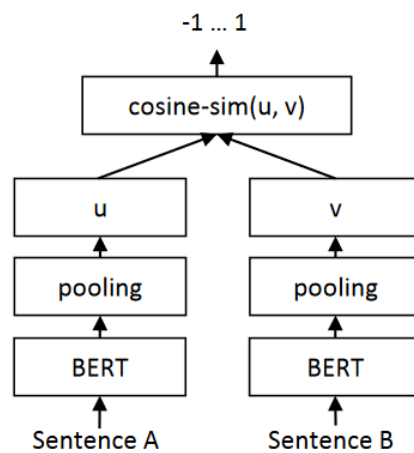


Figure 2.3: Sentence similarity using Sentence-BERT [22]

Another approach to generate sentence embeddings is the **Doc2Vec** model [23]. This method is an extension of the previously mentioned Word2Vec, but not only does it generate a vector for every word, but also a vector for each document (or sentence), called a **paragraph vector**. This way, each document also has its own representation and can be compared.

## 2.2 Tweet2Story

The work presented in this thesis is an **extension of Tweet2Story**, by Campos V. et al. [24]. This framework implements the automatic extraction of narratives from a set of *tweets*, but it doesn't work online, so these *tweets* have to be collected and pre-processed by the user. Our work aims to fill this gap by performing the automatic retrieval and cleaning of tweets for a topic of choice. We also extend this framework to the Portuguese language, since Tweet2Story focused on English *tweets*. The Tweet2Story narrative extraction pipeline, which will be a component of our work, is defined by these steps:

1. **Entity Extraction** - Retrieve entities (actors), using Named Entity Recognition (NER) models;
2. **Time Tagging** - Detect and extract temporal entities;
3. **Co-Reference** - Detect entities that refer to the same actor, and add them to the narrative if they weren't detected before;
4. **Event Extraction** - Detect and extract events (typically verbs);
5. **Extraction of Entity Relations** - Extract semantic relations between an actor and an event;

After extracting the narrative elements, this framework generates an annotation file that identifies the narrative, in accordance with the Text2Story annotation scheme, by Purificação et al. [25] (Figure 2.4).

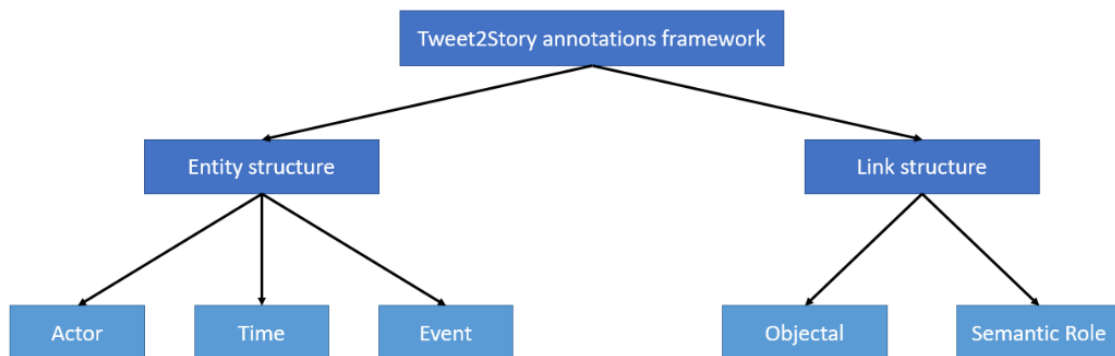


Figure 2.4: Annotation Scheme used by Tweet2Story [25]



## 2.3 Summary

In this chapter, we reported a set of concepts that help us gain context for the work that will be described afterwards. Some of the described **NLP** tasks, such as keyword extraction and will be used to construct the dataset described in Chapter 4, while the tasks of text similarity and summarization will be used in both Chapter 4 and 5. We also provided context on the Tweet2Story framework [24], which serves as a foundation of this work. To do this, we enumerated the main steps of its pipeline for narrative extraction and described the gaps that we address in this work.



## Chapter 3

# Related Work

Although Twitter posts contain a maximum of 280 characters, the volume of Twitter data is not to be underestimated. On average, 6000 tweets are posted per second [26], therefore, summarizing tweets is not a trivial task. In this chapter, we explore literature related to summarization of *tweet* streams and determining the relevance of a *tweet* to a given topic. Then, since we want to extend Tweet2Story to the Portuguese language, we also search for Portuguese datasets linking *tweets* to events or news articles, to find out if there are any datasets appropriate for our task. To guide our research, we divided it into three parts, that aim to answer the following questions:

1. What sort of work exists related to tweet summarization?
2. What sort of methods are used to calculate a score of relevance of a tweet?
3. Are there any existing Portuguese datasets linking tweets to news articles?

### 3.1 Discussion

#### 3.1.1 What sort of work exists related to tweet summarization?

In order to improve the performance of the Tweet2Story framework, the process of filtering and summarizing tweets in real time is crucial. There are many different methods used to perform the task of tweet summarization, such as graph-based [27, 28], cluster-based [29, 30], statistical approaches [27] and Integer Linear Programming (ILP) formulation [28, 31, 32]. Most of the literature found for tweet summarization focuses on extractive methods, where a subset of tweets is selected from a larger set. Also, there is more work regarding offline tweet summarization than evolving tweet streams.

Sumblr (2015) [29] is a framework that uses a clustering approach to group tweets, generates online and historical summaries and their respective timeline.

N. Alsaedi et al. (2016) [30] also propose a clustering-based approach while taking into

account the temporal dimension of the problem, by selecting tweets that have been a centroid of their cluster for a longer period of time.

The approach by A. Chellal, M. Boughanem (2018) [31] consisted of modeling the summarization task as an **ILP** problem, in which the goal is to maximize the summary relevance, while imposing restrictions regarding redundancy, coverage and temporal diversity.

The work of K. Rudra et al. (2015) [32] aims to help authorities gain situational information during crisis scenarios, in real-time. To achieve this, they begin by classifying tweets into situational or non-situational, and then use **ILP** to obtain the set of tweets that maximize the coverage of content words.

More recently, K. Rudra et al. (2019) [28] propose an extractive-abstractive approach to this problem. Firstly, since abstractive methods are less efficient, they select the most informative tweets in an extractive way, they generate paths between the bigrams of this smaller set to create new sentence. Like the previous work, they then use **ILP** to maximize the coverage of content words.

The work by Q. Li, Q. Zhang (2021) [27] proposes two methods for summarizing tweets. The first one is a semantic class based approach, in which they extract the semantic classes that answer to the questions that define an event: "Who?", "What?", "When" and "Where?", and they compute a score for each tweet based on those classes. The second approach uses a convolutional graph network, that calculates a salience score for each tweet from a tweet relation graph.

Work	Summarization		Tweet Stream		Approach
	Extractive	Abstractive	Offline	Real-time	
Shou L.[29]	X		X	X	Cluster
N. Alsaedi [30]	X			X	Cluster
A. Chellal [31]	X		X		<b>ILP</b>
K. Rudra [32]	X			X	<b>ILP</b>
K. Rudra[28]	X	X		X	Graph, <b>ILP</b>
Q. Li [27]	X			X	Statistic
Q. Li [27]	X			X	Graph

Table 3.1: Types of Tweet Summarization for different papers

As we can see from Table 3.1, abstractive summarization is still relatively new when it comes to the Twitter domain. Regarding extractive summarization, there is a wide variety of approaches such as clustering, **ILP**, statistical and graph-based methods.

### 3.1.2 What methods and metrics are used to determine the relevance of a tweet for a given event?

The task of determining the relevance of a *tweet* given an event is important for the creation of the dataset linking *tweets* to news articles. Some of the *tweet* summarization approaches mentioned above already assign a score to the *tweets* during their process.

Rishab S. et al. (2017) [33] compared three methods for the task that consisted in retrieving tweets with high recall and high precision: cosine similarity, the Okapi BM25 model and the Jelinek-Mercer smoothing. The BM25 function (Robertson and Zaragoza, 2009) [34], which obtained the best results out of these methods, estimates the relevance of a document for a given search query and ranks.

The work by A. Chellal. et al. (2017) [35] used the Extended Boolean Model to evaluate the relevance score of a *tweet*, by combining two scores, regarding the title and the description of the query.

Tao K. et al. (2012) [36] analyzed features, both syntactical and semantic, that can be used as indicators of a *tweet*'s relevance for a given topic. Their study suggested that features such as having a URL, being semantically related to the topic, or being a reply to another tweet, have a strong influence on whether a tweet is relevant.

Duan Y. et al. (2012) [37] proposed a timeline-based framework for topic summarization in Twitter in which they rank *tweets* according to the social influence of the users and the content quality, based on readability metrics.

### 3.1.3 Are there existing Portuguese datasets linking *tweets* to news articles?

One of the goals of this thesis was to use the Tweet2Story framework for a Portuguese approach, so we started by looking into Portuguese datasets related to news or *tweets* that could be used or adapted for our problem. The ideal dataset for our work follows this criteria:

1. Contains European Portuguese *tweets*;
2. Contains European Portuguese news;
3. The news are linked to related *tweets*;
4. Contains multiple different topics;

We found that not only was there a lack of datasets in the Portuguese language regarding news articles or *tweets* about events, but the few that existed include data written in Brazilian Portuguese, instead of European Portuguese.

The dataset by T. de Melo et al. [38] contains both *tweets* and news articles, having around 4

million *tweets* and 18 thousand news articles. However, the text is Brazilian Portuguese and it only contains data related to the Covid-19 pandemic in Brazil, and not general events.

Similarly, the QA-Corpus, by P. Cavalin et al. [39], is a Brazilian corpus that links *tweets* to news articles, but it is meant for question answering tasks. The *tweets* collected contain questions, and the news articles are used to answer them. Besides, this dataset concerns only the domain of Brazilian finance.

S. Moraes et al. [40] also created a corpus, Computer-BR, with the intent of evaluating different approaches to identifying subjectivity in Portuguese tweets about technology.

Dataset	Portuguese		Tweets	News	Topic
	European	Brazilian			
T. de Melo [38]		X	X	X	Covid-19
P. Cavalin [39]		X	X	X	Brazilian Finance
G. Brogueira[41]	X		X		None
S. Moraes [40]		X	X		Technology

Table 3.2: Types of Portuguese datasets with *tweets*

We found that not only was there a lack of work regarding datasets for the Portuguese language, but that there weren't any datasets that meet all our criteria. As we can see in Table 3.2, most of the datasets found are in Brazilian Portuguese, instead of European Portuguese, are focused on just one topic or don't link *tweets* to news articles. On that account, we defined the goal of creating a dataset that meets our needs, by collecting a set of events and related *tweets*.

## 3.2 Summary

In this chapter, we gained an understanding of existing research related to tweet summarization and narrative extraction. We found that although there was an abundance of work regarding tweet summarization, there isn't a standard method for this task. When it comes to datasets in the Portuguese language linking *tweets* to news, we couldn't find one that fit our criteria, and came to the conclusion that the existing datasets are either in Brazilian Portuguese, contain only *tweets* or only news, or are specific to a specific domain instead of having multiple events. This shortage of data motivated us to create our own Portuguese dataset linking news articles to *tweets* about different events.

## Chapter 4

# Tweet2Event-PT: Linking Events and Tweets in the Portuguese Language

Since our objective is to extract the narrative from a *tweet* stream, having a collection of news articles about world events is crucial, as they are our basis for retrieving the tweets we will work with. Upon exploring related literature and work in the last chapter, we came to the conclusion that there weren't any datasets linking *tweets* to news articles in the Portuguese language. That being the case, we opted to create a dataset befitting of our problem, which we named Tweet2Event-PT.

In this chapter, we describe our workflow in the creation of a Portuguese dataset with a set of events and their respective tweets. We implemented a method that can perform this task automatically, although we performed some adjustments manually in order to improve the overall quality of its data. This dataset and the implementation of this task are available on GitHub<sup>1</sup>.

The remainder of this section is organized as follows: Section 4.1 begins by explaining the methodology implemented for retrieving a collection of events associated with news articles. Next, in Section 4.2, we describe our process for collecting *tweets* related to the selected events. Then, in Section 4.3 we show how the *tweet* pre-processing pipeline was performed. Finally, in Section 4.4, we explain the experiments conducted to determine an effective method for selecting the most relevant tweets.

### 4.1 Event Retrieval

The first step for the construction of a dataset was selecting a set of events, preferably with associated news articles. We first considered an approach similar to A. Dusart et al. [42], who retrieved the summary of events from the Wikipedia "**Current Events**" portal<sup>2</sup>. This portal lists a collection of events that occur worldwide, and they are updated daily. However, working

---

<sup>1</sup><https://github.com/LIAAD/tweet2event-pt>

<sup>2</sup>[https://en.wikipedia.org/wiki/Portal:Current\\_events](https://en.wikipedia.org/wiki/Portal:Current_events)

with events from the whole world implied the risk of finding **more tweets written in Brazilian Portuguese**, rather than European Portuguese, since the Twitter Application Programming Interface (API) doesn't make a distinction between the two languages.

For this reason, and since our approach in this work is focused on the Portuguese language, it made sense to select events that were exclusive to Portugal. Upon looking for similar portals, we encountered the Wikipedia page "**2021 in Portugal**"<sup>3</sup>, which lists, in chronological order, a summary of events that happened in Portugal in 2021, with references to Portuguese news articles.

In order to retrieve the set of events in an automatic way, we looked for tools to help us extract the contents of a *Wikipedia* page, and found two Python libraries: *Pywikibot*<sup>4</sup> and *Wikipedia*<sup>5</sup>. *Pywikibot* was designed to automate work on *Wikipedia* and other *MediaWiki* pages, while the *Wikipedia* library was created with the focus of using already parsed data from *Wikipedia* pages.

Upon exploring *Pywikibot*, we noticed that we could only retrieve the page content (see Figure 4.1) in **Wikipedia's source-code notation** (see Figure 4.2), which meant we would have to manually parse this code in order to obtain the data we needed. In the case of the *Wikipedia* library, it was possible to retrieve the content of the page already parsed, but there wasn't a way to get the **Uniform Resource Locator (URL) of the references** used for each event. That being the case, we opted to use *Pywikibot* for this task.

## Eventos [ edit | edit source ]

---

### Janeiro [ edit | edit source ]

- 1 — Tem início a [Presidência Portuguesa do Conselho da União Europeia](#), que se estende pelo primeiro semestre de 2021.<sup>[1]</sup>
- 4 — [Dia de luto nacional pelas exéquias de Carlos do Carmo, na Basílica da Estrela](#).<sup>[2]</sup>

Figure 4.1: Wikipedia "2021 in Portugal" page event example

Using regular expressions, we were able to clean the text and retrieve the dates, summary of events and the URLs of the news articles used as reference.

Then, we proceeded to perform keyword extraction using Yet Another Keyword Extractor (YAKE) [14] on the extracted summaries, so they could be later used as the search query for retrieving related *tweets*. Each of these keywords was then manually curated to guarantee their appropriateness in terms of Twitter retrieval. By following these steps, we were able to generate a Comma-Separated Values (CSV) file with the following fields, for 58 events throughout the year 2021:

- **ID** - Identifier of the event, in chronological order;

<sup>3</sup>[https://pt.wikipedia.org/wiki/2021\\_em\\_Portugal](https://pt.wikipedia.org/wiki/2021_em_Portugal)

<sup>4</sup><https://github.com/wikimedia/pywikibot>

<sup>5</sup><https://github.com/goldsmith/Wikipedia>



---

```

== Eventos ==
=== Janeiro ===
* [[1 de janeiro|1]] -- Tem início a [[Presidência do Conselho da União Europeia
|Presidência Portuguesa do Conselho da União Europeia]], que se estende pelo primeiro
semestre de 2021.<ref>{{citar web|url=https://www.publico.pt/2021/01/01/politica/
noticia/portugal-preside-quarta-ue-1944747 |título=Portugal preside pela quarta vez
à UE |publicado='''[[Público (jornal)|Público]]''' |data=1 de janeiro de 2021|
acessodata=1 de janeiro de 2021}}</ref>
* [[4 de janeiro|4]] -- [[Luto nacional|Dia de luto nacional]] pelas exéquias de
[[Carlos do Carmo]], na [[Basílica da Estrela]].<ref>{{citar web|url=https://www.
publico.pt/2021/01/01/culturaipsilon/noticia/carlos-carmo-governo-decreta-dia-luto-
nacional-segundafeira-1944761 |título=Carlos do Carmo: Governo decreta um dia de
luto nacional para segunda-feira |publicado='''[[Público (jornal)|Público]]''' |data=
1 de janeiro de 2021 |acessodata=1 de janeiro de 2021}}</ref>

```

---

Figure 4.2: Wikipedia "2021 in Portugal" page source-code notation example

- **Topic** - Keywords of the event, to be used as a query for tweet searching;
- **Summary** - Description of the event taken from the page;
- **Date** - Date of the event;
- **URL** - Links to the news articles used as reference;
- **News Date** - Publishing date of the news article used as reference;

These events will not be listed here since their summaries are large, but this file is available in our repository<sup>6</sup>.

## 4.2 Tweet Retrieval

Having a set of events, we now had to retrieve *tweets* related to each of them. To achieve this, we used the *Tweepy*<sup>7</sup> Python library, for accessing the Twitter **API**. This **API** allows the retrieval of *tweets* from the complete history of public *tweets* matching a given search query.

For the search query, we used the query defined in the previous section, with keywords extracted from **YAKE** that were then manually curated, and added some operators to filter our search, as shown in Figure 4.3.

Firstly, we added a language filter to obtain only *tweets* written in Portuguese, but it is to note that Twitter uses best-effort method to detect the language. Then, we added an operator to remove *retweets*, since we wanted to avoid duplicates. When it comes to replies, we decided to keep them since they can be used to separate what is meant to be a longer post, and can contain

<sup>6</sup><https://github.com/LIAAD/tweet2event-pt>

<sup>7</sup><https://docs.tweepy.org/>

---

```

query = "(" + topic + ") -Brasil -brasileiro -brasileira -br \
-is:retweet -place_country:br lang:pt"

tweets = client.search_all_tweets(query, max_results=100,
start_time=start_time, end_time=end_time, tweet_fields="created_at")

```

---

Figure 4.3: Query used to retrieve tweets for a given topic

useful information as well. Since some events are already discussed in social media before being reported, we set a time window of seven days before its date until one day after the date of the event.

In one of our first experiments, we noticed that the output contained *tweets* in Brazilian Portuguese that were related to Brazilian events. For example, regarding an event about Portugal winning a futsal championship, we obtained tweets about another futsal match with Brazil that happened some days before. To minimize these numbers, we added operators to exclude words such as "*Brazil*", "*Brazilian*" or the abbreviation "*br*".

In total, we were able to obtain 52,422 *tweets* over 58 events, which gives us an average of 903 *tweets* per event. Upon inspection of the resulting tweets, we noticed that some events contained very few *tweets*, ranging from 2 *tweets* to a maximum of 15,476. We also realized that some events contained many *tweets* with a strong opinionated nature instead of an informative one. Hence, we decided to select a subset of these events, in order to work with *tweets* that are more relevant to their respective topic, originating a dataset with 12 events and 28,315 related tweets in total. Table 4.1 lists the number of *tweets* for each of these events.

Event	Number of Tweets
Start of general confinement	1820
Decision for Operation Marquês	15476
Running over with vehicle with Eduardo Cabrita	313
Arrest of Joe Berardo	553
Arrest of Luís Filipe Vieira	2218
Fire in Castro Marim	187
Fire in Odemira	99
Death of Jorge Sampaio	3590
Portuguese futsal team wins world cup	995
Election of Rui Costa for SLB	827
Soccer Match: Benfica vs Belenenses	1567
Arrest of João Rendeiro	670
Total	28315

Table 4.1: Number of tweets per event before pre-processing

### 4.3 Tweet Cleaning

After examining the collected tweets, it was necessary to clean them so we could proceed to the Natural Language Processing (NLP) tasks. Because of Twitter posts informal language and special marks, these pre-processing steps are different than they would be with a formal text. For example, *tweets* include emojis, hashtags, URLs and mentions to other users, e.g. "@user". The script for this process is available on our GitHub page, allowing the replication of the dataset. Taking this into account, we followed the following steps to clean our tweets:

1. Eliminate **emojis**, using a regular expression;
2. Strip redundant **whitespace characters**;
3. Remove **links of attachments** at the end of *tweets*;
4. Delete **user mentions** at the beginning of *tweets*;
5. Remove **user mentions** and **hashtags** at the end of *tweets*;
6. Delete **exact duplicate tweets**;
7. Eliminate *tweets* with **less than 7 words**;
8. Remove *tweets* with **high similarity** (over 80%) using TF-IDF cosine similarity.

After removing duplicate and similar tweets, our dataset now has a total of 22,371 tweets, and the number of tweets by topic can be seen below, in Table 4.2.

Event	Tweets
Start of general confinement	1419
Decision for Operation Marquês	12937
Running over with vehicle with Eduardo Cabrita	222
Arrest of Joe Berardo	369
Arrest of Luís Filipe Vieira	1200
Fire in Castro Marim	128
Fire in Odemira	51
Death of Jorge Sampaio	2511
Portuguese futsal team wins world cup	858
Election of Rui Costa for SLB	767
Soccer Match: Benfica vs Belenenses	1443
Arrest of João Rendeiro	466
Total	22371

Table 4.2: Number of tweets per event after pre-processing

## 4.4 Tweet Relevance

After performing the *tweet* cleaning and text pre-processing pipeline, we were left with filtering the resulting *tweets* in order to keep only the most relevant ones. The best way to do this would be to manually annotate the relevance of the *tweets* to an event, but it would require a large effort. To circumvent this problem, we opted to rely on the Signal dataset [43], which is manually annotated with the relevance of a tweet to a news article, thus fulfilling our objective, despite being in English. Such dataset, gives us the chance of testing and experiment a few methods regarding the process of filtering out non-relevant tweets. The learned method will then be applied in our framework. In this section, we detail these experiments and their respective results.

### 4.4.1 Experiments on the Signal Dataset

The "**Signal-1M Related Tweets**" dataset contains around 6,000 tweets linked to news articles. Each entry in this dataset contains a **topic** field, which is an identifier for a news article in the "**Signal 1 Million News Articles Dataset**" [44]. Each *tweet* is also labelled with its relevance to its respective news article, that can be classified as:

- 0: **Not relevant** to the news article
- 1: **Somewhat relevant** to the news article
- 2: **Highly relevant** to the news article

After we performed the pre-processing and filtering steps, as explained before, the resulting dataset contained a total of 4737 *tweets* linked to 97 topics, giving us an average of 48.8 tweets per news article. By taking a look at Figure 4.4, we can see the distribution of tweets by topic. The topic with the largest amount of *tweets* contains 67 of them, while the smallest contains only 20, so the dataset isn't very balanced.

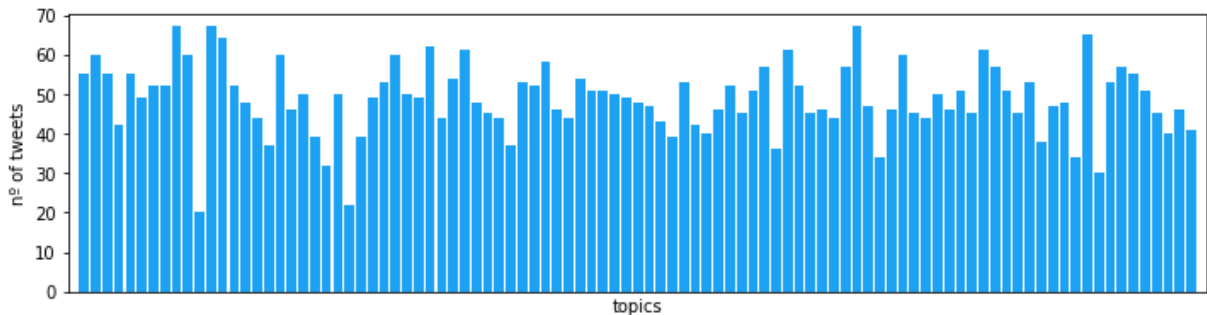


Figure 4.4: Distribution of *tweets* by topic on Signal dataset

Regarding the distribution of tweets by relevance, as we can see from Figure 4.5, the vast majority of tweets in this dataset are classified as not relevant to the news article, and the number

of tweets classified with **1** and **2** are very close.

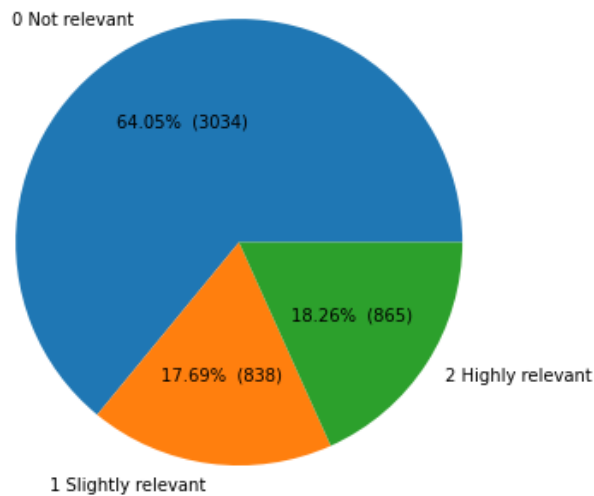


Figure 4.5: Distribution of tweets by relevance

Having a better idea of how this dataset looks like, we proceed to the task of filtering non-relevant tweets. To achieve this, we tried a variety of lexical and semantic methods anchored on an Information Retrieval (IR)-approach, where *tweets* which are considered to be more similar to the query of a topic are pushed to the top, whereas most dissimilar ones are pushed to the bottom. To generate a query for each topic, we extracted the top 10 keywords from its respective news article. The five methods we tried to determine relevant *tweets* to the issued query are as follows:

1. **Okapi BM25** [34], ranking function, based on Term Frequency–Inverse Document Frequency (TF-IDF), used by search engines to estimate the relevance of documents to a given search query;
2. **YAKE relevance** measure, adapted from R. Campos et al. [45], in which a headline corresponds to a *tweet* in our work;
3. Cosine similarity between **SentenceBERT** [22] embeddings, using the pre-trained BERT model;
4. Cosine similarity between **Doc2Vec** [23] embeddings, with the following parameters:
  - Epochs: 100
  - Vector dimension: 20;
5. Cosine similarity between **average Word2Vec** [21] embeddings, using the Google News pre-trained model.

To evaluate the results, we used the evaluation software used in the Text REtrieval Conference (TREC), which allows the use of many metrics to compare the results of a document retrieval

task with a ground truth. The methods we tried assign a continuous score to each document (*tweet*), instead of a boolean score, therefore, we cannot compute metrics such as precision or recall directly. Additionally, the relevance values of the dataset have 3 classes, so for this evaluation, we considered as relevant tweets both the ones with a relevance score of **1** and **2**.

For this reason, we began by using the **Normalized Discounted Cumulative Gain (NDCG)** metric, a function that computes the quality of a ranking. As we can see from Figure 4.6, **BM25 ranks the highest** and it's the only one with a score higher than 0.6 (a perfect ranking would have a **NDCG** score of 1), while the remaining methods have similar scores to each other.

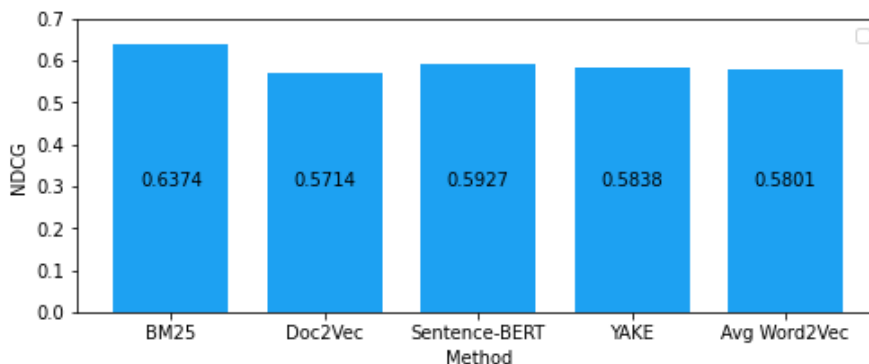


Figure 4.6: **NDCG** results

We also applied **top-based measures**, that is, metrics that compute a certain metric assuming that the top  $K$  results are the retrieved documents, such as **Precision@K**, **Recall@K** and **F1@K**.

Since this dataset has an average of almost 50 tweets per topic, we calculated Precision@K and F1@K for  $k \in \{1, 5, 10, 20, 30, 40, 50\}$ . Looking at the **Precision-K curve** in Figure 4.7, **BM25 surpasses** other models but it's also the one whose precision decreases the most as  $k$  increases. Given that and each topic has an average of 48.8 *tweets* and 64% of those are not relevant, it's natural that precision has a low score for  $k = 50$ , since it means we're retrieving all documents for a large portion of the topics.

Finally, we plotted a **Precision-Recall Curve** to understand the trade-off between precision and recall, for each threshold of 0.1 in recall. As we can see from Figure 4.9, the **Okapi BM25 function excels** when compared to the word embeddings methods and the **YAKE** measure, although its results aren't the best. It's interesting to see that the remaining models provide very similar results, although **none was particularly good at the task**.

#### 4.4.2 Experiments on Tweet2Event-PT

Given that BM25 obtained the highest **NDCG** and precision scores in the last section, we adopted it as our function to rank the tweets we collected by relevance.

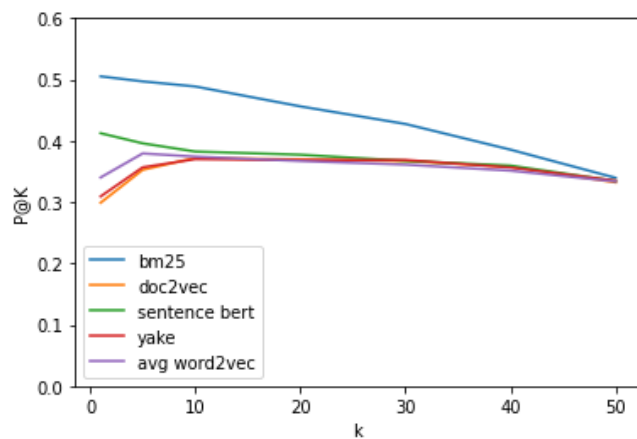


Figure 4.7: Precision@K of different methods on the Signal dataset

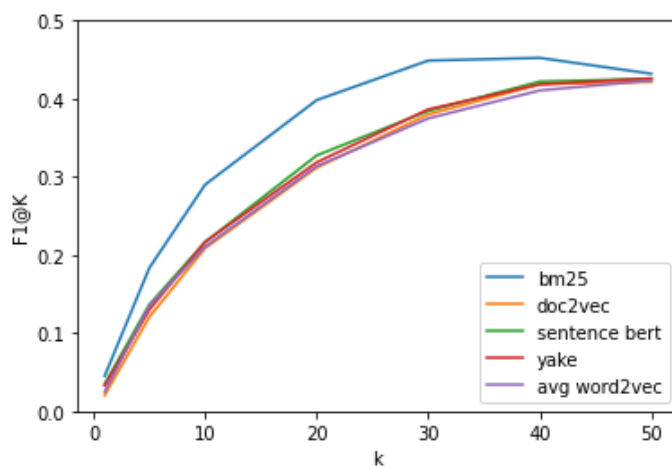


Figure 4.8: F1@K of different methods on the Signal dataset

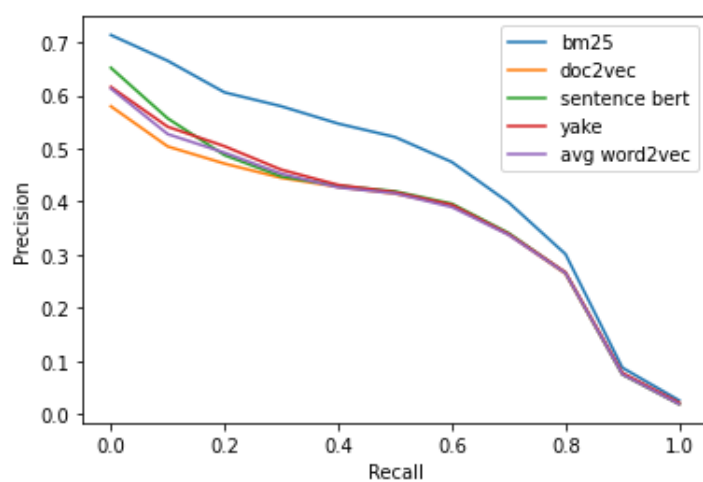


Figure 4.9: Precision-Recall Curve of different methods on the Signal dataset

Our approach for applying the Okapi BM25 function is very similar to the one we used in the Signal dataset. First, we used **YAKE** to retrieve the top 10 keywords for each of the news articles. Then we used these keywords as the query for the BM25 function, and the *tweets* for each event as the corpus. After that, for each event, since this function is unbounded, we normalized the *tweet* scores, so they would use the same scale of 0 to 1, where 1 would be the normalized score of the highest-ranking tweet of each event, similar to the approach by W. Jiexin et al. [46].

As mentioned previously, since our dataset was quite large, it would require a significant effort to manually annotate all the tweets with their relevance. Therefore, in order to validate the results of the BM25 function in our dataset against a ground truth, we extracted a balanced sample and manually evaluated a small portion of it.

To create this sample, we divided each event's tweets into 10 bins with intervals of 0.1 in the BM25 score. Then, we performed a random sample of tweets with 5 tweets from every bin in order to have a balanced sample. To evaluate their relevance, we had three annotators from our lab who labelled their relevancy by attributing a score of 1, if relevant, or 0, if not relevant to its topic. Since some bins contain fewer than 5 *tweets*, in total, we annotated 473 *tweets* over 12 events.

All that was left now was to define a threshold for separating relevant and irrelevant *tweets*. This threshold could have been experimented with in the Signal dataset, but since the distribution of *tweets* of the two datasets is very different, the threshold should be specific to our dataset. Let  $T = \{0, 0.01, 0.02, \dots, 0.99\}$  be a set of thresholds, in which tweets with  $score \geq t_i \in T$  are considered relevant, and tweets with  $score < t_i \in T$  are irrelevant. We computed the precision and recall of these documents against the documents with our manual evaluation for each of these thresholds. It is important to have a large enough quantity of tweets, so we found that having a threshold of 0.2 was a reasonable compromise between the quantity of relevant *tweets* and the precision, having around 25% of relevant tweets. Figure 4.10 .

Having determined this threshold, the dataset now contains a total of 6471 relevant *tweets* distributed over 12 events, ranging from 45 to 2543 *tweets* (Table 4.3).

## 4.5 Summary

In this chapter, we implemented a method for retrieving events that occurred in Portugal and collecting related *tweets*, for the purpose of extracting narratives from them. Firstly, we obtained a set of 58 events from the Wikipedia page "**2021 in Portugal**". Then, using their keywords, that we extracted with **YAKE** and manually curated, we retrieved likely related tweets using the Twitter **API** and selected a subset of the most relevant events. To improve the quality of the text, we performed some data cleaning tasks, such as deleting **URLs**, mentions and duplicate or highly similar documents.

Furthermore, we tested out different methods for ranking them by relevance on the "**Signal**



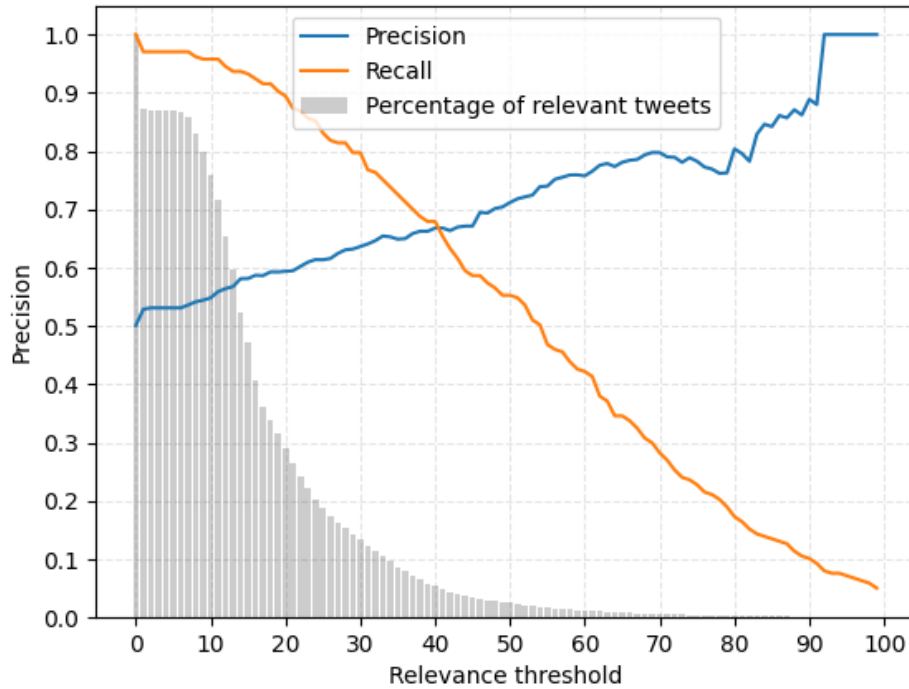


Figure 4.10: Precision, recall and percentage of relevant tweets for different BM25 scores thresholds in the selected sample

Event	Tweets
Start of general confinement	232
Decision for Operation Marquês	2543
Running over with vehicle with Eduardo Cabrita	73
Arrest of Joe Berardo	194
Arrest of Luís Filipe Vieira	77
Fire in Castro Marim	64
Fire in Odemira	45
Death of Jorge Sampaio	1096
Portuguese futsal team wins world cup	508
Election of Rui Costa for SLB	686
Soccer Match: Benfica vs Belenenses	784
Arrest of João Rendeiro	169
Total	6471

Table 4.3: Number of relevant tweets in the dataset

**1M Related Tweets**" dataset. After comparing the results of the different functions, we applied **Okapi BM25** in our **Tweet2Event-PT** dataset, and validated the results using a sample of our dataset as the ground truth.

To conclude the process, we defined a BM25 threshold for considering a tweet relevant, and filtered the dataset as to keep the ones with the highest scores. In the end, our dataset contained around 6500 tweets linked to 12 events with references to news articles.

## Chapter 5

# TweetStream2Story: Narrative Extraction from Tweets Collected in Real-Time

This chapter describes the pipeline of creating the TweetStream2Story framework, which allows the extraction of narrative elements from Twitter, in both real time, or from past events, and its visualization along a timeline.

This framework was built on top of existing services such as the Twitter Application Programming Interface (API), Elasticsearch and Text2Story tools<sup>1</sup> for extracting narrative elements. We will firstly describe the general architecture of this framework, and then proceed to detail its main components

### 5.1 Use Case

The main use case for this application can be defined as follows:

**Given a query about a topic/event and a time period, automatically collect tweets related to it, extract their narrative, and allow its visualization.**

The main goal for this framework was retrieval of tweets in real-time, as an event is ongoing, but ended up extending it to support the collection of past tweets. As suggested by the use case diagram (Figure 5.1), the framework also stores each topic's data, allowing an user to view multiple topics simultaneously. Regarding the visualization of a topic's narrative, we opted to use a timeline that lets us view the narrative for each time window of the event. Each time window has a narrative associated to it, that can be viewed through a knowledge graph, accompanied by the tweets that generated it.

---

<sup>1</sup><https://text2story.inesctec.pt/>

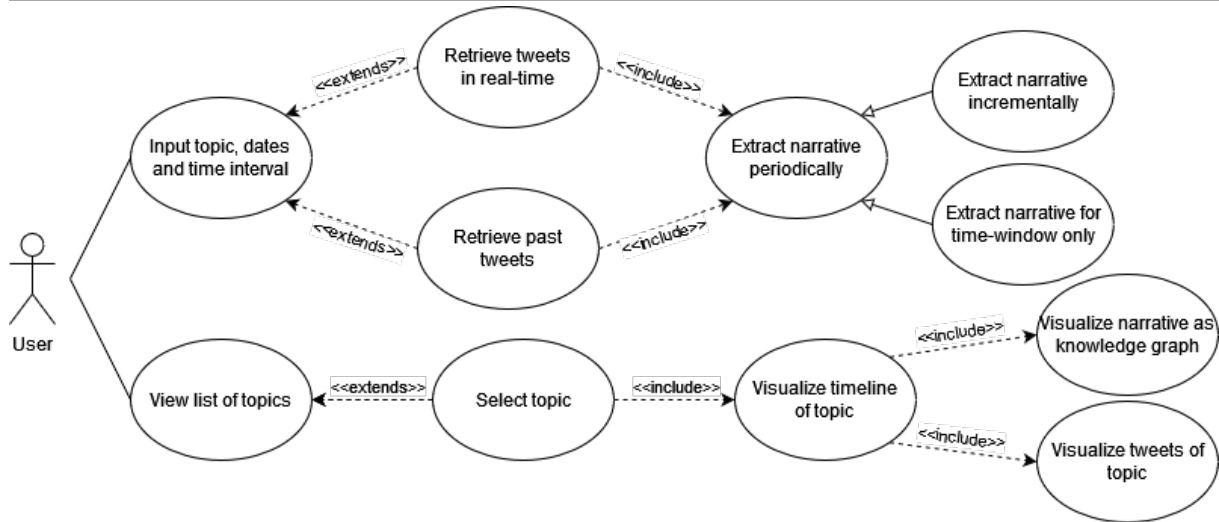


Figure 5.1: Use Case Diagram

## 5.2 Architecture of the Solution

In a simplified way, our framework consists on collecting *tweets*, summarizing them, extracting their narrative, and visualize it. To accomplish this, we need to make use of a variety of services, and the architecture of our solution must take into account the connections between these different services.

Firstly, we use the **Twitter API** to **collect tweets** and their metadata for a given query, both for streaming and for past *tweets*. Then, we must use a database that allows the **efficient storage and retrieval** of these *tweets*. Since our previous experiments showed that the **Okapi BM25 function** was a good choice to select relevant tweets given a query, we decided to use it as our **summarization** method. To accommodate both storage and summarization, we used **ElasticSearch**, a document oriented **database** that uses BM25 as its main function for retrieving documents. Next, to **extract the narrative** elements from the collected tweets, we used the **Text2Story framework**. Finally, in order for a user to **visualize the results** and input topics, we used React to create a **web page** as our client.

To connect all these services, we created a **web server** using **Flask**, a minimal Python framework that can be easily extended. This framework allows us to build a Representational State Transfer (**REST**) **API** for communicating with the client and serves as the middle point for retrieving tweets, storing them in the database and extracting their narrative, as shown in Figure 5.2.

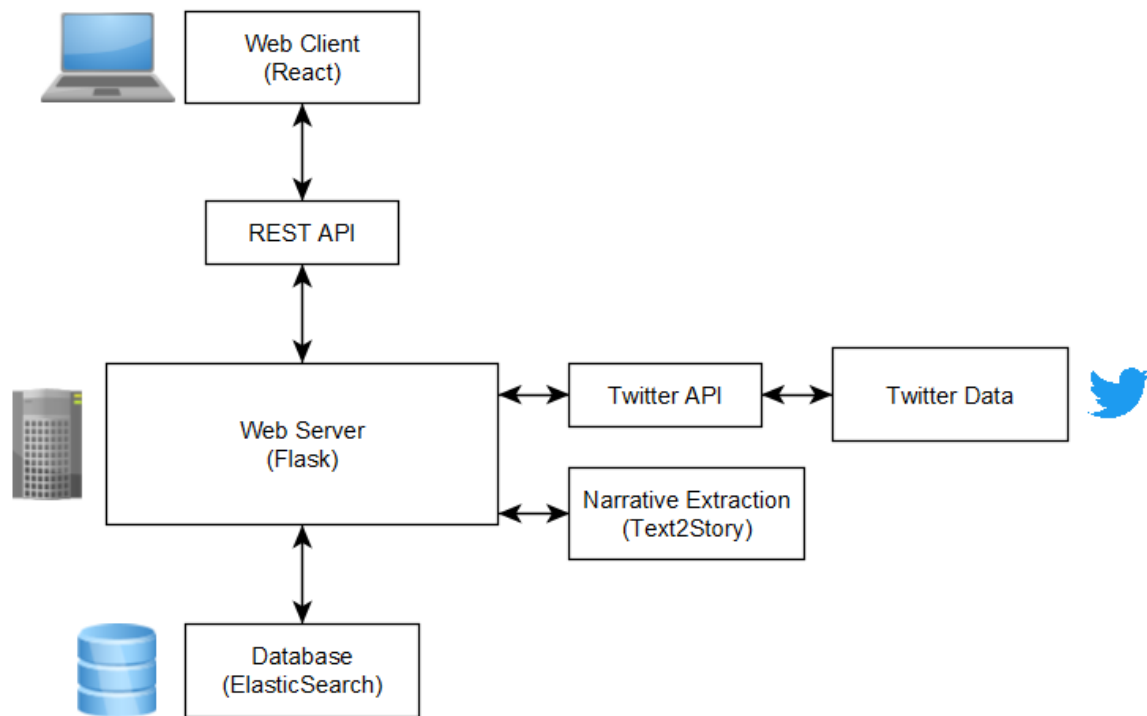


Figure 5.2: TweetStream2Story Architecture Overview

### 5.3 Elasticsearch

As mentioned in the previous section, we decided to use Elasticsearch as our database. Elasticsearch is a search engine that works with all kinds of data, and its distributed nature, speed and scalability motivated us to use it for this demo. This engine also allows the use of different functions to retrieve documents based on a query, and uses the BM25 function as the default relevance algorithm.

The first step in this pipeline was *mapping*, which is the process of defining how a topic and its fields are stored and indexed. In Elasticsearch, an index is a collection of documents, and each document is a collection of key-value pairs containing our data, serialized as JavaScript Object Notation (**JSON**) objects. These documents can be schema-less (meaning that Elasticsearch will infer new fields as they are added and map them to their respective types) or they can have a dedicated structure.

Defining our own mappings brings us many advantages such as distinguishing between short and full-text strings, working with consistent date/time formats, and it optimizes operations of searching and indexing new documents.

That being the case, the first step for our storage pipeline was to define the mappings for our data. Firstly, we created an index that stores the metadata for all topics created by a user, named **topics**. Each document in this index contains fields such as the start and end date of

the *tweet* retrieval, the time interval for generating narratives, whether a topic is active, and its respective time windows.

To store the tweets, each topic will have its own index, identified by the id of the document on the "topics" index. In each topic's index, each document corresponds to a tweet, and will contain its original text, the pre-processed text and the time it was created.

After defining the structure of our indices, in order to secure this database, we added Hypertext Transfer Protocol (HTTP) Basic Authentication (BA). This way, every request sent to ElasticSearch must be sent from our web server, where the credentials are stored.

## 5.4 Flask Web Server

After setting up the database, we needed to create the application that communicates with and the other services. For this step, we created a Flask application, which acts as the center of our framework. Flask is a Python micro web framework, since it doesn't require particular tools or libraries, and can be easily extended to accommodate the functionalities we need. This framework allows us, for example, to define a set of routes for building a REST API to communicate with the client. It also allows us to import other tools that help us collecting tweets, cleaning them and scheduling the extraction of their narrative.

In Figure 5.3, we can see the directory of this Flask application.

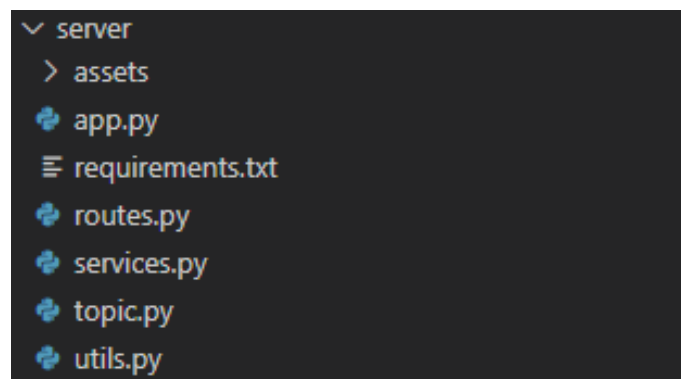


Figure 5.3: Directory structure of flask application

The file **app.py** is in charge of loading the Flask application and of starting the threads for processes running in the background. When the application is initialized, the topics are loaded from ElasticSearch into memory.

The file **services.py** initializes the services used by the app, such as Elasticsearch and the Twitter API, as well as processes that will be running in the background, such as Python's scheduler<sup>2</sup>. It is to be noted that this scheduler has some limitations, as it can only schedule

<sup>2</sup><https://schedule.readthedocs.io/>

The file **routes.py** contains all the routes used to define the endpoints of our **REST API**.

Next, the **topic.py** file contains the definition of the Topic class, which will be detailed in the next section.

The file **utils.py** has miscellaneous functions, such as the text cleaning pipeline or calculation of the similarity between documents in the same topic.

Lastly, the **assets** folder contains miscellaneous files, such as a dictionary of emoticons, that will be used in the text cleaning steps.

### 5.4.1 REST API

In order for the user to communicate with the server, we built a **REST API** that returns the necessary data in **JSON** format. This **API** contains endpoints for adding, retrieving or deleting topics, as well as starting and stopping the collection of tweets, as shown in Table 5.1.

Endpoint	HTTP Method	Function
/api/topics	GET	Get metadata of all topics
/api/topic	POST	Create a new topic
/api/topic/<id>	GET	Get metadata of topic with given <b>id</b>
/api/topic/<id> /window/<window_index>/tweets	GET	Get global and interval tweets of the <b>n</b> -th window of topic with given <b>id</b>
/api/start_topic/	POST	Start collecting tweets for topic with given <b>id</b>
/api/stop_topic/	POST	Stop collecting tweets for topic with given <b>id</b>
/api/delete_topic/	DELETE	Delete topic with given <b>id</b>

Table 5.1: TweetStream2Story **REST API** Endpoints

For accessibility and ease of implementation, we didn't want to make it a requirement for the user to log in in order to use our application. However, we still wanted to give users privacy such that they can only view the topics added by themselves. With this in mind, we decided to add Token-based authentication to our API as a means of identifying a user, in which the token will be generated by the client and persist in local storage after accessing the demo web page.

### 5.4.2 Topic Class

All the operations of our framework revolve around the topic inputted by the user, so we implemented a class to define it. The life-cycle of a topic is defined by the following steps:

1. Creation of the topic
2. Start of the retrieval of tweets

3. Retrieval of tweets
  - Clean the tweet
  - Store it
  - At the end of each time window, update the similarity matrix
  - At the end of each time window, extract the narrative of window
4. Finish the tweet retrieval
5. Deletion of the topic

## Attributes

To obtain a better understanding of the attributes of the **Topic** object, Table 5.2 provides a description of each field and their type.

Attribute	Type	Description
tag	String	Query given the topic
start_date	Datetime	UTC Date in the format YYYY-MM-DDThh:mm:ssZ
end_date	Datetime	UTC Date in the format YYYY-MM-DDThh:mm:ssZ
time_interval	Integer	Period for generating narrative
time_unit	String	Time units for <b>time_interval</b> (e.g. "minutes", "hours", "days")
active	Boolean	Topic is currently collecting tweets
finished	Boolean	Topic has finished collecting tweets
twitter_id	String	ID for the rule of respective stream
es_id	String	ID of the respective index in ElasticSearch
time_windows	Array	Narrative of each time window
mode	String	Indicates mode for retrieving tweets (streaming, past search)

Table 5.2: Attributes of class Topic

## Methods

**start()** - This method is called by the scheduler (using Python's schedule library) at the object's **start\_date**. It builds a Twitter rule for this topic and adds it to the Twitter stream. The query of this rule consists of a disjunction between the keywords given as input. It also filters out retweets and only retrieves tweets in the topic's language, using Twitter's language detection, which may not be always accurate.

**e.g.** (soccer OR Portugal OR Spain) -is:retweet lang:en

If this step is successful, the function then schedules the function **add\_window()**, which, after each time window ends, will extract its respective narrative, using the **time\_interval** and **time\_unit** parameters.



**stop()** - Removes the topic's Twitter rule from the tweet stream, cancels the job scheduled to extract the narrative and marks the topic as finished. It should be noted that a topic cannot be resumed after stopping it.

**results(i, isGlobal)** - This function retrieves from Elasticsearch the top 50 *tweets* (block 5.1 for the *i*-th time window of this topic, in one of these two modes: global or interval.

```
results = es.search(  
    index = id,  
    body= {  
        "query": {  
            "bool": {  
                "must": [ {"match": {"text": query}} ],  
                "filter": [ {"range": {"date": dateQuery}} ]  
            }  
        }  
    },  
    size = 50  
)
```

Listing 5.1: Querying Elasticsearch for top 50 tweets

If the *isGlobal* parameter is set as True, the date range corresponds to the topic's start date until the end of this time window. Meanwhile, if we want the results in the interval mode, the date range corresponds to the period between the start and ending of this time window only. This way, the global modes provide us with a vision of everything that's happened so far in an event, while interval mode gives us more information of a shorter period of time. After obtaining the results from Elasticsearch, we sort them by chronological order and then return them.

**add\_window()** - Called by the scheduler every time a time window has ended. This function begins by retrieving the top 50 tweets for the time window that has just finished, for both global and interval mode, using the function **results()**. After obtaining the top tweets, they are joined in a single string, with a full stop separating them, as to prepare them for the extraction step. Then, this text is sent to the Text2Story framework through a socket, which will return a **JSON** object containing the extracted actors, events and content of the annotation file.

Having the narrative object of the time window in global mode, we update it by adding a new entry containing actors that weren't present in the previous time window. This way, when generating the respective knowledge graph, we can color the nodes of the new actors with a different color, so the user can easily perceive which information is new.

**retrieve\_past\_tweets(key, secret)**

This functions retrieves the tweets for a topic with the **past mode**, using Twitter's API. The query used is build in the same way specified in **start()**.

This mode has different requirements compared to collecting tweets in real time. The Twitter API allows a maximum of 300 requests per 15 minutes for the full-archive search. For this reason, it's not viable to collect tweets in this mode with our credentials, since we would have to find a way to handle multiple topics and users at once. For this reason, it is necessary that the user inputs their own Twitter credentials (consumer key and consumer secret) when looking up past events. However, these credentials are **not** stored in Elasticsearch, meaning they are discarded as soon as the topic is created.

## 5.5 Webpage

For the visualization component of this framework, we developed a webpage where users can test our framework, using React<sup>3</sup>, a JavaScript library.

The main components of this webpage are:

- An area to **input a new topic** and its respective parameters;
- A **list of the topics** the user has added, shown when the webpage is loaded;
- **Visualization of a topic's narrative** along a timeline.

### 5.5.1 Token Generation

As mentioned in Section 5.4.1, in order for a user to create and interact with their own topics, requests must contain a token that identifies the user. To do this, we generate a Universally Unique Identifier (**UUID**) upon loading the webpage, using the `uuid`<sup>4</sup> package, which generates cryptographically-strong random values. This value then is stored in the browser's local storage, and sent as an authentication token in every request to our **API**.

### 5.5.2 Topic Input

The first step for generating the narrative regarding an event is choosing it and its options, so the topic input is the first component the user sees. After typing in a topic and clicking on "**Extract Narrative**", a modal opens where the user can provide the necessary parameters, such as the Twitter API credentials, the desired language, the duration of each time window, as shown in Figure 5.5.

The user can also select the mode they wish to use for collecting tweets, which determines if it'll use the Twitter Streaming API or the Search API. If the user selects "**Past Event**" or "**Schedule**" they can set the the start and end date in which the tweets were posted. If they

---

<sup>3</sup><https://reactjs.org/>

<sup>4</sup><https://www.npmjs.com/package/uuid>

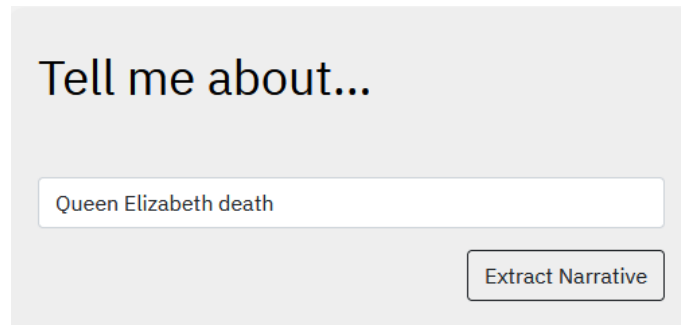


Figure 5.4: Query input for topic

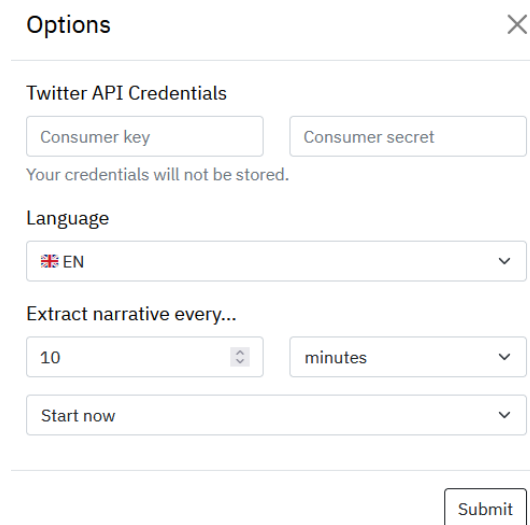


Figure 5.5: Options for narrative extraction

are looking up an **ongoing event** but don't know when to stop the search, they can choose the "**Start now**" option and stop the tweet retrieval when they wish (Figure 5.6).

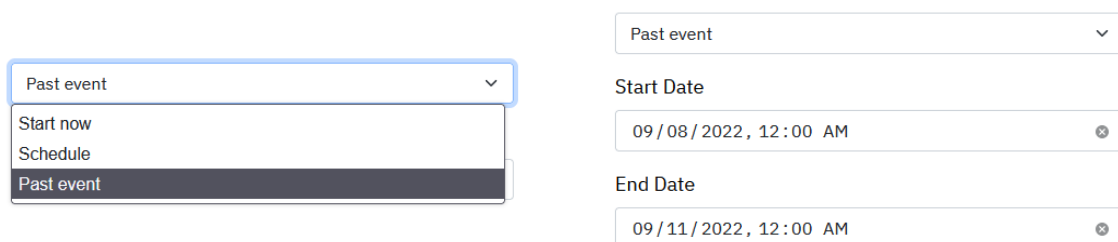


Figure 5.6: Tweet retrieval mode

### 5.5.3 List of topics

After submitting a new topic, it will be added to the list of topics by the user, where they can view the status of a topic, visualize the narrative of a topic or do actions like stopping the retrieval of tweets or deleting a topic.









Active Topics						
#	Topic	Language	Start Date	End Date	Status	Options
1	seleção portugal espanha	 PT	2022-06-02 19:45	2022-06-02 22:15	 Completed	 Open 
2	Queen Elizabeth death	 EN	2022-09-11 15:27	2022-09-10 12:00	 In progress...	 Open 

Figure 5.7: Topics list

#### 5.5.4 Topic Visualization

By opening one of the topics shown in the list above, the user can then see the **timeline** of that event. They can choose between the two modes we mentioned: **incremental/global view** or **interval view**. In each mode, the marks on the timeline correspond to the periodical time windows defined previously, and each one shows the narrative and information of that time window.

The default visualization of the time window is the knowledge graph, where the nodes represent the actors, and the edges are the semantic relations between them (Figure 5.8).

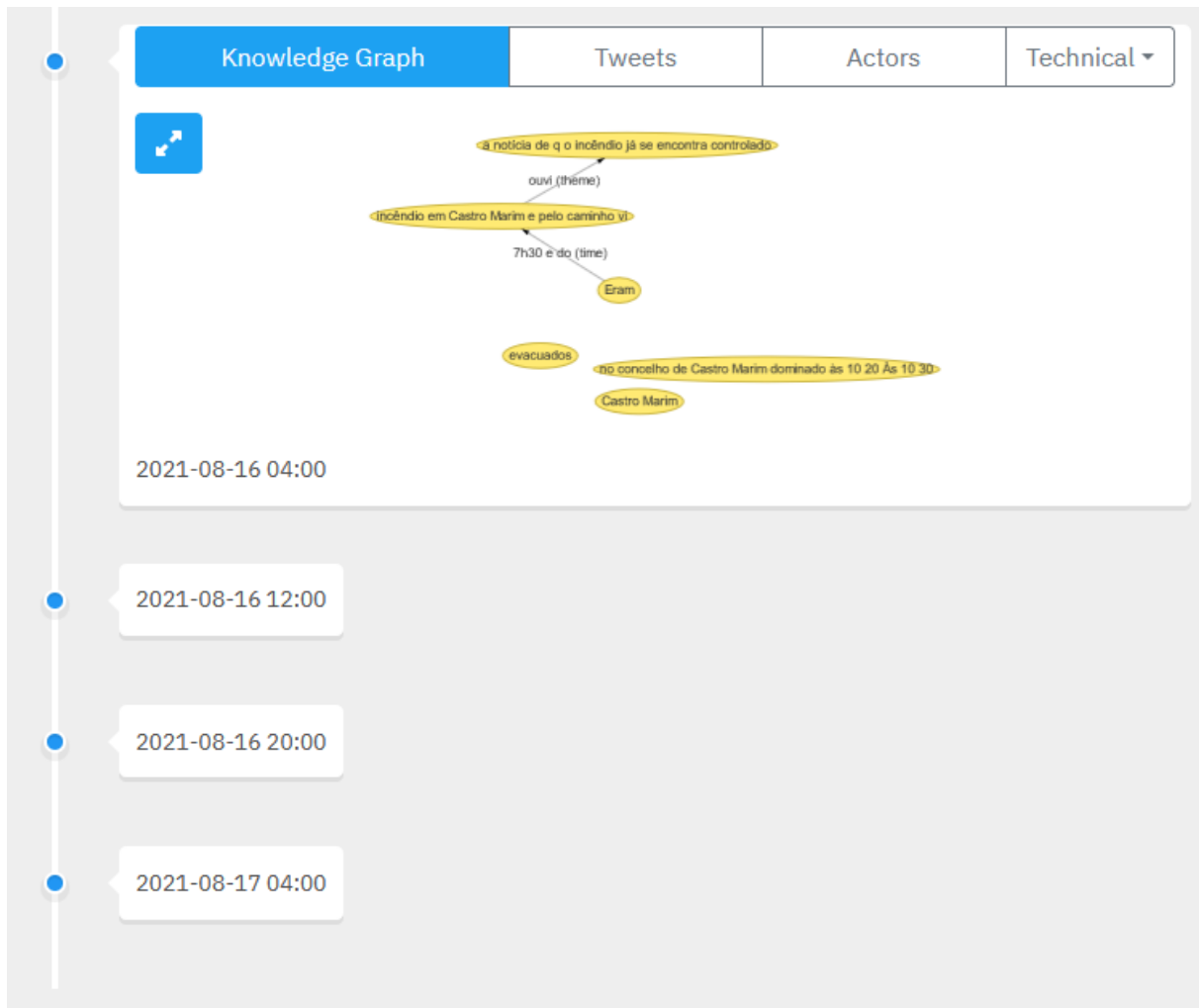


Figure 5.8: Timeline interface

As said before, each time window’s annotation contains a list of actors that are new, compared to the previous time window, so that we can represent them easily in the knowledge graph. In Figure 5.8, all the nodes we see are yellow since they are in the first time window of this event, and therefore all information is new to the user. However, in the next time window (see Figure 5.9), we see as blue the nodes shown previously, and as yellow the ones.

This visualization can be switched to the list of tweets that were used to generate the narrative, or a list of the actors. For more advanced users, who want to conduct research, for example, they can also view and **download** the information in other more formal representations such as **Discourse Representation Structures (DRS)** or the Text2Story annotation [25].

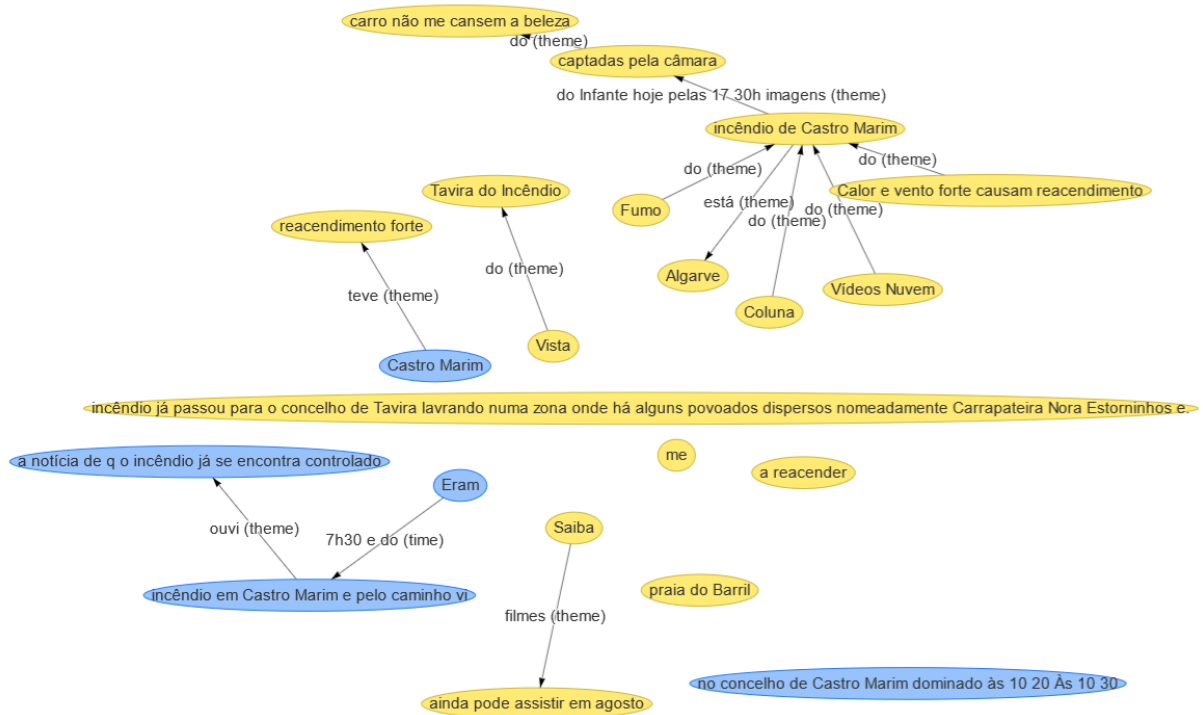


Figure 5.9: Knowledge graph with different colors to represent new actors

## 5.6 Summary

In this chapter we described our process for creating a framework that allows the visualization of narratives from *tweets* collected in real time. This framework consists of a web server created with Flask, that connects to other services like Elasticsearch, for storage, the Twitter API, to retrieve tweets, the Text2Story framework, for extracting narratives, and a client for visualization of the results.

We began by configuring Elasticsearch and defining the structure of the documents that will be stored. Then, we developed a Flask application that implements a **REST API** and methods to collect *tweets* and scheduling the narrative extraction for each time window of an event. Finally, for the component visualization, we created a webpage with React, where the user can input multiple topics and visualize the narrative of a topic of their choice. The topic can be visualized through a timeline, where each window can be expanded to display the narrative knowledge graph and the tweets used to generate it.

## Chapter 6

# Results and Discussion

In this chapter, we conduct an evaluation of our approach for extracting narratives from a stream of *tweets* in real time. We start by describing our methodology for evaluating the results of this work, as well as the difficulties and limitations we encountered.

In Chapter 4, we've already conducted some experiments to assess the effectiveness of Okapi BM25, the function we opted to use for summarization, in filtering relevant tweets from irrelevant ones. However, we have yet to evaluate the overall results of our framework. The representation of the results in our approach depends on many factors such as the **volume and quality of tweets** posted at a certain time, the **summarization method**, and the **narrative extraction pipeline**. Given that the narrative extraction pipeline has a substantial influence in the knowledge graphs and is not one of our contributions, it is not trivial to evaluate the results.

Consequently, we opted to obtain feedback from potential users through **two different surveys**: one for evaluating the **usability and visualization** representations of the demo, and another to evaluate the **quality of the information** in the knowledge graphs.

### 6.1 Usability and Visualization Survey

In the first survey, we intend to gather information about the usefulness and usability of the TweetStream2Story framework. This survey is divided into two sections. As this was a short survey, we obtained a total of 211 responses. The full results can be viewed in Appendix A.

The first section is comprised of questions regarding their age, academic qualifications, habits for obtaining information through different media and their usage of Twitter, as a means to characterize the potential users of this framework. The second section consists of a set of statements about different types of visualization of narratives, with which the user can agree or disagree. More specifically, this survey had the following goals:

- Determine the types of users likely to use this framework;

- Determine if Twitter is seen as a reliable platform for keeping up with events;
- Understand the most commonly used media for obtaining news;
- Determine the usefulness of different visualizations such as the timeline, knowledge graph, set of tweets;
- Understand if a knowledge graph of a narrative can complement or replace a news article;

### 6.1.1 Discussion

As we can see from the results in Figure A.3, the majority of the respondents use social media more often than other sources for news, with around 125 votes selecting it as the most frequently used.

By grouping the data by age range (Figure 6.1) we can also see that in all age groups, social media is the one with the majority votes for the most used media, except for the eldest group (above 40 years old). This data is in accordance to our assumption that social media is outgrowing other news formats.

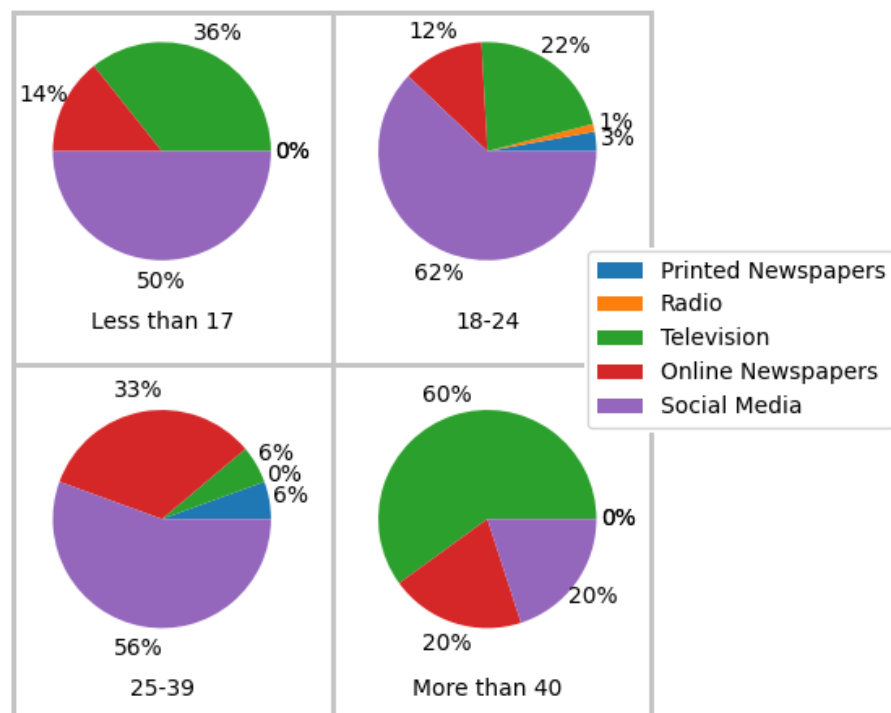


Figure 6.1: Most used media for obtaining news by age group

We obtained a total of 211 responses, and the vast majority of the respondents were in the age group of 18-24 years old (see Figure A.1). Regarding academic qualifications, most of the people surveyed have completed high school, and there is a significant portion with a Bachelor's degree A.2.



Regarding the means for finding news, most of the respondents marked social media as their most frequently used media for obtaining news. We mentioned in Chapter 1 how social media has been a very valuable tool for journalists, and the answers to this question show how important social media is for readers as well. When it comes to usage of Twitter, we found that around **72% of the respondents use it**, and most of these have used it to **keep up with an event** (see Figure A.5).

As for the second section of this survey, which aims to evaluate the representation of the results, we obtained mixed feedback. About the use of the timeline to understand an event over time, the results reveal an almost **even distribution** of opinions, and we believe this may be caused by the fact that the timeline isn't interactive in the survey. Most of the users agreed that **listing the tweets** selected to generate the narrative is a **useful** feature, but there are very **mixed opinions** regarding whether the knowledge graph **simplifies** a narrative compared to a list of tweets.

Finally, regarding the knowledge graph, we provided a knowledge graph about a fire that occurred in Castro Marim, Portugal in 2021, as well as a news article about it [47]. Most voters agreed that it complements the news article, but there is disagreement on whether it can replace the news article, even with a short amount of time.

## 6.2 Narrative Assessment Survey

In this second survey, our objective is to perform a deeper evaluation of the narrative results. This survey was quite extensive when compared to the first one so, in total, we obtained only 19 responses. The results can be seen in Appendix B. First, we selected **three events**, listed in Table 6.1, and for each of these, asked the respondents some questions about the journalistic elements present in the extracted relations, which ones are irrelevant or interesting. For two of these events, the *tweets* used are **Portuguese**, while the remaining event uses **English tweets**. For the Portuguese language, events were collected from our dataset and indexed in Elasticsearch. Meanwhile, for the remaining English events, we retrieved its tweets in real-time using our framework (see Table 6.1).

Event	Language	Source
Fire in Castro Marim, Vila Real de Sto. António and Tavira	Portuguese	Our dataset
Soccer Match: Benfica vs Belenenses	Portuguese	Our dataset
Shooting in Denmark mall	English	TweetStream2Story

Table 6.1: Events used for evaluation through survey

For this survey, we established the following goals:

- Determine whether a user can understand the information in the knowledge graph;

- Determine which extracted entities provide interesting information not included in news articles;
- Determine if there is a gain of information from a time window to the next one;

The first question for each of these case studies asked users to identify what journalistic elements ("who", "what", "where" and "when") were present in the actors and relations captured in a knowledge graph. For each actor and relation, we annotated which of these journalistic elements were present in them. Then, we compared their answers to our ground truth in order to evaluate the users' perception of the extracted narrative.

Since each actor/relation can have more than one element (for example, they can mention both time and location), the user score for one actor/relation can be seen as the recall of their answers relative to our ground truth. For example, if the correct answer for an actor/relation is ["What", "Who"] and the user selects only ["Who"], their score for this actor/relation is 0.5 (see Figure 6.1, where  $U$  is the set of user answers, and  $G$  the set of the ground truth answers). Then, the user score for the entire event is calculated as the average of all their entity scores (see Equation 6.2, where  $E$  is the set of actors/relations).

$$score_e = \frac{|U \cap G|}{|G|} \quad (6.1)$$

$$user\_score = \frac{\sum_{e \in E} score_e}{|E|} \quad (6.2)$$

In Table 6.2, we can see the average score of the 19 respondents for each event, and the average score of the three events. At a first glance, we can tell that the higher the number of actors/relations in the graph, the lower the user scores, but these entities will be further inspected in the next subsections.

Event	No. of actors/relations	Average user score
Fire in Castro Marim	27	64.6%
Soccer Match: Benfica vs Belenenses	37	47.6%
Shooting in Denmark Mall	29	50.6%
Average score for all events		54.27%

Table 6.2: Overall user scores

Next, we aggregated the results for each of the four journalistic elements ("who", "what", "where" and "when"). For each of these, we calculated how many the users got correct in average for each event. Tables 6.3 and 6.4 show the average scores of all users, accompanied by the total number of these elements present in the extracted actors/relations. The easiest elements to identify were there "Where" elements, followed by "Who", "What" and "When". In Twitter, users usually talk about what's happening in the present, and this timestamp is already shown in the

metadata of the tweet. For this reason, the extracted "When" elements are very few, as they only mention past events, so users may have been hesitant to mark them. The "What" elements also have low scores, as it is the element that occurs more times.

Event	Who		What	
	Avg. Score	Total	Avg. Score	Total
Fire in Castro Marim	94.7%	1	48.9%	10
Soccer Match: Benfica vs Belenenses	38.8%	19	38.5%	19
Shooting in Denmark Mall	53.9%	4	43.7%	13
Average score for all events	62.42%		44.9%	

Table 6.3: Overall user scores for "Who" and "What" elements

Event	Where		When	
	Avg. Score	Total	Avg. Score	Total
Fire in Castro Marim	62.8%	15	36.8%	1
Soccer Match: Benfica vs Belenenses	100%	0	15.8%	2
Shooting in Denmark Mall	47.4%	19	42.1%	4
Average score for all events	70.07%		31.57%	

Table 6.4: Overall user scores for "When" and "Where" elements

In the remainder of this section, we take a closer look at the results of each of these events, by providing examples of extracted actors and relations considered interesting, as well as irrelevant or confusing ones. The timeline of these events can be visualized in our demo<sup>1</sup>.

### 6.2.1 Case Study 1: Fire in Castro Marim

The first case study concerns a fire that ignited in Castro Marim, Portugal, on August 16, 2021, that extended to the cities of Tavira and Vila Real de Santo António. For this event, we used the query "Incêndio Castro Marim Tavira Vila Real de Santo António" to retrieve the top tweets with Elasticsearch.

This event had a large number of relevant tweets and was able to generate useful and interesting relations between actors. The narrative extraction was able to, for example, provide us with data about the number of firefighters on scene, which is valuable information in order to have a better notion of the dimension of a fire disaster.

**"aos 357 bombeiros no local"**  
"to the 357 firefighters in the place"

<sup>1</sup><https://tweetstream2story.inesctec.pt>

Although the query already mentions three different counties, we were also able to extract smaller locations where the fire was getting close to, such as Carrapateira, Nora and Estorninhos.

**"Fogo do incêndio já passou para o concelho de Tavira lavrando numa zona onde há alguns povoados dispersos nomeadamente Carrapateira Nora Estorninhos"**

"The fire has passed to the county of Tavira, reaching a zone where there are some dispersed villages, namely Carrapateira Nora Estorninhos"

While in general, the extracted actors and relations are relevant to the event, others are confusing or irrelevant. For these cases, one interesting example is:

**"Expansão do incêndio 584 96 0"**

"Expansion of the fire 584 96 0"

The original tweet used emojis next to the numbers to represent what they were counting, instead of words (584 firefighters, 196 rescue vehicles). With the removal of emojis in the tweet cleaning phase, this information got lost, although it was originally useful. Another example of an irrelevant relation extracted is:

**"Caos destaques TSF"**

"Chaos TSF Highlights"

This relation was extracted from a *tweet* posted by a radio station company (TSF) highlighting some ongoing events such as this fire and the **chaos** in another country's airport. However, the word **"destaques"/"highlights"** shouldn't be connecting **"Caos"/"Chaos"** and **"TSF"**.

Out of the three events we studied, we believe this one provides more informative and clear results, which may be caused by the fact that the tweets used are also informative and are less speculative. In the evaluation of the respondents identification of journalistic elements, this event obtained the highest score (64.6%), as shown in Table 6.2), when averaging the score of all users, confirming that users also found it easier to understand.

Regarding the use of time windows to observe the evolution of a topic over time, we compared two consecutive time windows with the global narrative: 4am-12pm and 4am-8pm of August 16, 2021. The actors and relations captured in the second time window provide us new information about the fire, such as its re-ignition caused by the strong wind, or that it has extended to other cities. However, there are some actors not relevant to the topic, like loose prepositions.

### 6.2.2 Case Study 2: Soccer Match: Benfica vs Belenenses

This event covers a soccer match between two Portuguese teams during the pandemic. One day before the match, a Covid outbreak was detected in the Belenenses team, but the game wasn't

rescheduled. Belenenses (B-SAD) then played with only 9 members, no substitutes, and the game ended at the beginning of the second half, with a 7-0 score. For this event, we used the query "SLB Benfica Belenenses B-SAD", since people refer to the teams by their acronyms most of the time.

While the previous event captured mostly informative *tweets*, since it deals with an emergency situation, this event has a more controversial nature, therefore, the *tweets* collected contain more personal opinions. Some examples of this include pitying or blaming the teams, such as in:

**"Coitado do B-SAD"**

"Poor B-Sad"

**"Vergonha"**

"Shameful"

However, the knowledge graph is still able to capture information about the number of players and their role, like in:

**"SAD vs Benfica - Belenenses SAD entra com apenas 9 jogadores 2 deles guarda redes"**

"SAD vs Benfica - Belenenses SAD enters with only 9 players 2 of them goalkeepers"

The framework also captured information unique to this match, explaining why it wasn't postponed, for example:

**"O Belenenses SAD entrevista a dizer que não queria adiar"**

"Belenses SAD interview saying they didn't want to postpone"

When it comes to confusing elements, one example is this one:

**"avanzado uruguaio Santa Clara Boavista Barcelona B SAD"**

"uruguayan striker Santa Clara Boavista Barcelona B SAD"

The *tweet* where this actor came from mentioned that one of the players, Darwin, scored the 4th brace (two goals in a single match) of the season. Then the tweet lists the teams (Santa Clara, Boavista, Barcelona, B SAD), but they were all captured as the same actor.

When it comes to the users' perception of the knowledge graph through journalistic events, this particular event had the lowest average score. From the responses, we can see that there in general there wasn't a consensus about which journalistic elements were present in the extracted relations, and in many of them, some users didn't mark any element, which means they considered it irrelevant or were confused by it. We agree that this event had the most confusing narrative, which is likely a consequence of its controversial nature, causing many users to use offensive language and producing *tweets* of lower quality. Despite these worse results, the framework was still able to capture information about the match as well as the public's view of it.

### 6.2.3 Case Study 3: Shooting in Denmark

This last event deals with a shooting that happened on this year's 4th of July, in a shopping centre in Copenhagen, that resulted in 3 deaths and 4 injured people. For this event, we used the query "Denmark Shooting" to retrieve tweets.

When asked about interesting information about this event that wasn't contained in the summary, most users pointed out these relations, which contain information about the last shootings:

**"Denmark had one shooting back in 2015"**  
**"That was their first major shooting in Denmark in years"**

On the same day, another shooting happened in Illinois, USA, so the knowledge graph shows us comparisons between the two shootings. The following example, for example, comparing the number of shootings between Denmark and the USA, was the most voted as interesting:

**"I think the gun laws may have something to do with these statistics"**

The knowledge graph was also able to capture statistics about the number of injuries caused, like in the following relation:

**"DENMARK 3 dead 3 critically wounded in shooting at Denmark mall"**

There is also some interesting information present, regarding the suspect of the shooting, that wasn't captured well by the knowledge graph. For example, one of the tweets mentions that the suspect was held in a psychiatric facility, but the actor "**suspect**" wasn't extracted, and instead the relation generated was:

**"Denmark held in psychiatric facility"**

Below follows another example of a confusing relation, although it makes sense grammatically, apart from the case in "The Independent". In this case, "The Independent" is the name of a news agency and was only written at the end of the original *tweet*.

**"dead after shooting The Independent"**

The average user score for this event was 50.6% (see Table 6.2), falling between the other two events, and the journalistic element that appears the most is the location, which is to be expected since "Denmark" and "Copenhagen" appear in many actors. When it comes to the use of time windows, we showed the user the time windows between 4:30pm-6:30pm and 4:30pm-8:30pm. In the second one, many of the new actors/relations were considered as relevant, with 5 out of 12 actors/relations being selected by more than 30% of the users (see Appendix B).

### 6.2.4 Discussion

In this section, we analyzed three case studies, each pertaining to a different domain: a natural disaster, a sports event and a mass shooting. In all of these events, the extracted narrative was able to capture useful data and interesting information that wasn't present in the summaries of these events.

We tried to evaluate the user's understanding of these knowledge graphs by measuring how much journalistic elements they identified correctly, as mentioned in Section 6.2. The case study with the highest average user score was "Fire in Castro Marim", then "Shooting in Denmark", and the one with the lowest score was "Soccer Match: SLB vs Belenenses". The first case study lacked user opinions, while the third one had a lot of subjective information due to the controversial nature of the match. This shows us that events with controversial opinions generally generate more confusing *tweets* and content that isn't informative, and that the selection of *tweets* has room for improvement.

When it comes to the use of a timeline, we can conclude that diving the narrative into time brings relevant information windows allows us to better understand the evolution of an event through time.

Finally, for the three events studied, there was a consensus from the users about extracted elements that were interesting and were not included in the summary provided to them, like specific locations or comparisons with other similar events that happened shortly before. Therefore, we can conclude that the knowledge graphs generated by our framework bring novel information that can complement a news article.

## 6.3 Summary

In this chapter, we conducted an evaluation of our framework. Since the results of our platform are heavily influenced by the narrative extraction pipeline, we found it **difficult to formally evaluate** them, as they are not a direct consequence of our contribution. For this reason, we conducted two different surveys, with the aim of evaluating the usability and the quality of the knowledge graphs, respectively.

In the first survey, we learned that social media was a valuable platform for obtaining news not just for journalists, but for casual users as well. We also concluded that users find the knowledge graph a **good complement to a news article**, but there is disagreement on whether it can act as a replacement for it.

In the second survey, we performed a deeper evaluation of the elements present in the knowledge graph for three different topics, two in Portuguese, and another in English. From the results and our assessment, we concluded that Twitter posts go beyond the information present in news articles and that the narrative extracted complements them. However, the results still

contain irrelevant or confusing information, that can be caused by many factors such as the text cleaning pipeline or the extraction algorithms.



## Chapter 7

# Conclusions

In this chapter, we will start by recapitulating the main ideas of this thesis and present the conclusions drawn from each chapter. Then, we will discuss the limitations of this work, whether and how we can improve them in the future, as well as possible additions to the Tweet2Story framework.

In the beginning of this document, we demonstrate how Twitter is a valuable resource for journalists who want to gather information for their articles. We also mention that this thesis aims to **extend the existing Tweet2Story project**, by Campos V. et al. [24]. This framework implements the automatic extraction of narrative elements from a bundle of tweets, but it only works offline. The main goal of this thesis is to improve it through the **retrieval of tweets in real-time**.

Since the narrative extraction tool has a high computation time, and taking into account the large dimension of Twitter data, we knew that only a small portion of tweets could be used for each extraction. Then, the first step of this work was to research about **existing approaches for tweet summarization**, so we would work with a small set of *tweets* of higher quality. Similarly, we also performed some research in order to find **datasets in the Portuguese language** that fit our task, but concluded that there is very scarce research in this area.

This knowledge then led us to the next phase of our work, described in Chapter 4: a creation of a **dataset in the Portuguese language** linking tweets to events. To obtain a set of events, we referred to the Wikipedia page "**2021 in Portugal**", which lists events that happened in Portugal in chronological order, with a brief summary and references to Portuguese news articles. By using some tools to parse this page, we were able to store these events as a dataset containing their **date, summary and references**. Then, by choosing a query for each of these events, we **retrieved related tweets** using the Twitter Application Programming Interface (API).

After the collection of tweets, we performed some **text pre-processing** tasks and **eliminated similar entries** to improve the quality of the data. Then, we performed an experiment on another dataset, and found that the **Okapi BM25** was the best method for **ranking tweets by relevance**. Finally, we determined a threshold for this function in order to distinguish relevant

and irrelevant *tweets* in our dataset, Tweet2Event-PT.

Next, we implemented TweetStream2Story, a **framework for generating narratives from a set of tweets in real-time**. The demo of this framework allows a user to input a query about a topic and choose parameters such as the period for collecting tweets, language and the duration of each time window. The framework then starts retrieving tweets (either as they are being posted, or *tweets* from the past), and while they're being collected, generates the narrative for each time window using the most relevant tweets. The user can then visualize the resulting narrative through a knowledge graph in a timeline, accompanied by the tweets that generated it and the respective annotations.

Lastly, we were left with evaluating the results of our work. Since the main product of this thesis was a demo so we conducted **two surveys**: one for the **visualization and representation** of the results, and another for the **quality of the results**. Regarding the visualization of the results, we came to the conclusion that a knowledge graph is a good complement to a news article, and that listing the *tweets* used is useful for the user to inspect the results in more detail. As for the quality of the extracted narratives, we determined that the *tweets* posted by users can bring novel information not present in other media like news articles. However, further improvements can be done in order to reduce irrelevant narrative elements as well.

In conclusion, we believe TweetStream2Story is able to capture the narrative of stories told by Twitter and that it can be very beneficial not only for journalists who seek to obtain information as it is posted, but to users who want to stay updated on a topic of their choice.

## 7.1 Limitations

One of the main contributions of this thesis was a **Portuguese dataset** linking tweets to news. Since the datasets in the Portuguese language are very scarce in the Natural Language Processing (NLP) domain, the dataset we created can be very useful in a variety of tasks. However, we would need a lot of time and effort to manually annotate the relevance of each *tweet* in it. For this work, we annotated a smaller sample of the dataset in order to compare the performance of some methods to a ground truth, but the scores obtained may not necessarily reflect the real results.

Regarding the **TweetStream2Story demo**, the main limitation is the need for a user to input its **Twitter API credentials** for retrieving past tweets. Although our objective was to make this demo accessible to anyone, these credentials are required in order to programmatically access Twitter data. One solution could be using a single pair of credentials for all users, but it isn't very feasible because of the constraints of the Twitter **API**, especially without knowing the dimension of the user base.

## 7.2 Future Work

Although the work of this thesis has contributed to the improvement of the Tweet2Story project, there are still many possible directions we can follow, involving a wide variety of areas. Regarding the **dataset** we created and curated, it would be useful to manually **annotate all tweets with their relevance**, so that its entirety could be used as the ground truth in supervised document retrieval tasks. A further improvement would be to manually annotate the narrative of the retrieved events, so we could formally compare our knowledge graphs to a ground truth.

As for the **summarization** of the retrieved *tweets*, it would be interesting to implement offensive speech and irony detection techniques, to filter out some *tweets* and improve the quality of the results. We would also like to test and apply **abstractive approaches** in order to generate original content, instead of using only sentences that come directly from Twitter users.



# Appendix A

## Results of Usability Survey

Results obtained from the survey conducted to evaluate the usefulness and usability of our framework.

### A.1 Section 1

1. What age group do you belong to? (in years)

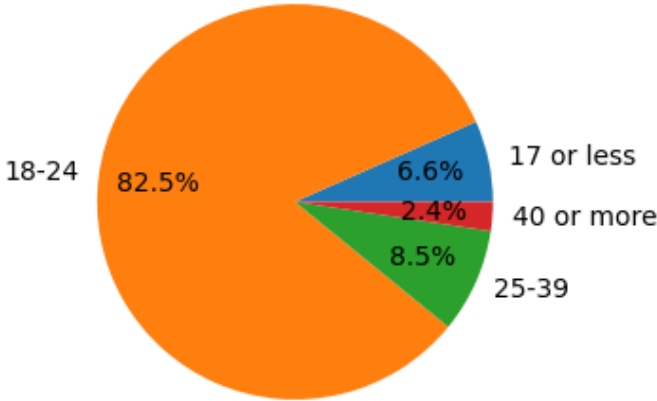


Figure A.1: Age

## 2. What are your academic qualifications?

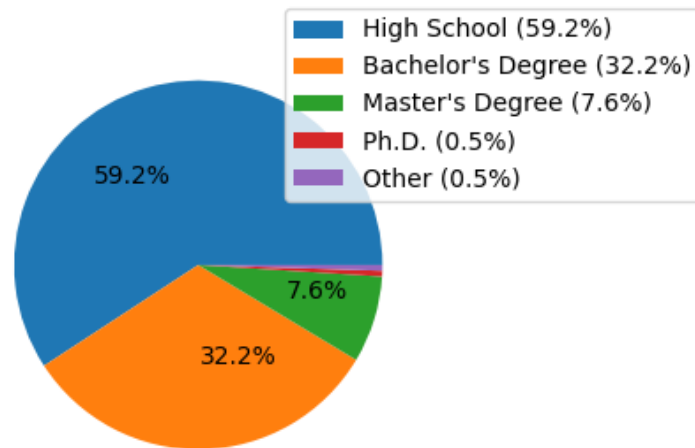


Figure A.2: Academic Qualifications

## 3. Which of these media do you use the most to obtain news? Order them from 1-5, where 1 is the least frequently used, and 5 the most frequently used?

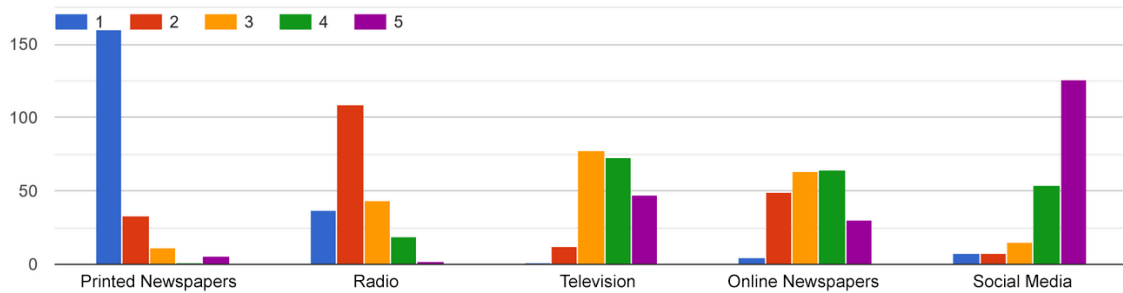


Figure A.3: Ranking of preferred media for obtaining news

#### 4. Do you use Twitter?

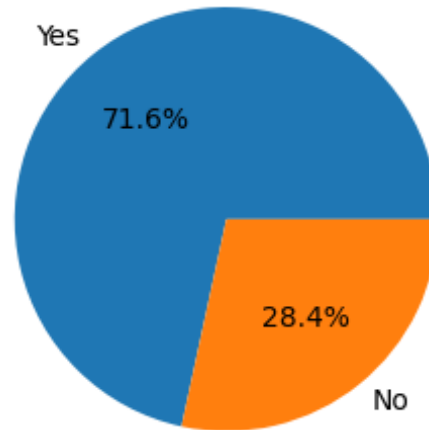


Figure A.4: Twitter Use

5. If you are a Twitter user, how often do you use it to keep up with events in real-time through a series of *tweets*? (Where 1 corresponds to "Never", and 5 to "Very frequently")

161 responses

From the results of this question, we conclude that our platform can be of interest for the majority of the respondents, since over 55% of responses were 4 or 5.

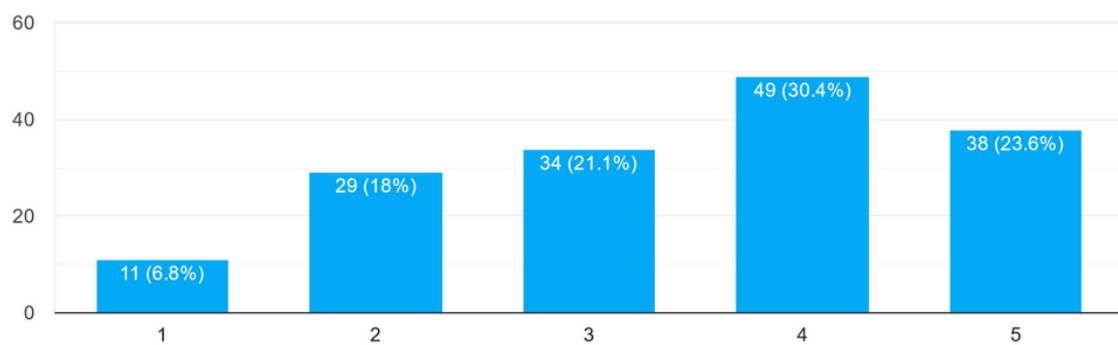
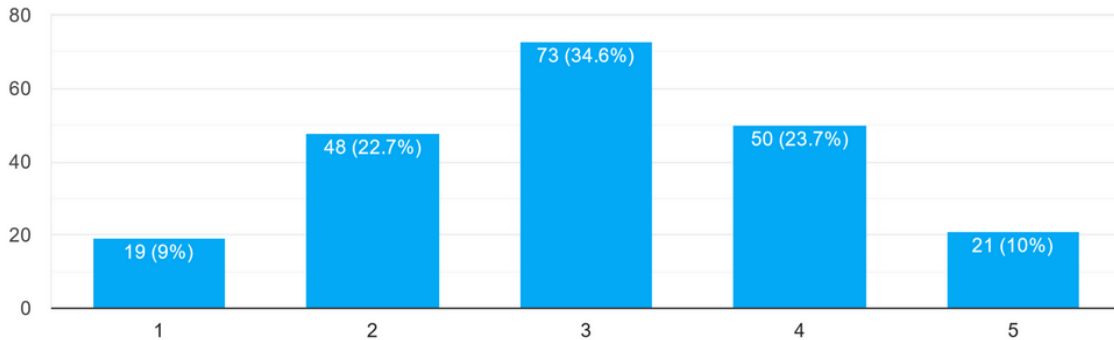


Figure A.5: Frequency of using Twitter to keep up with events in real-time

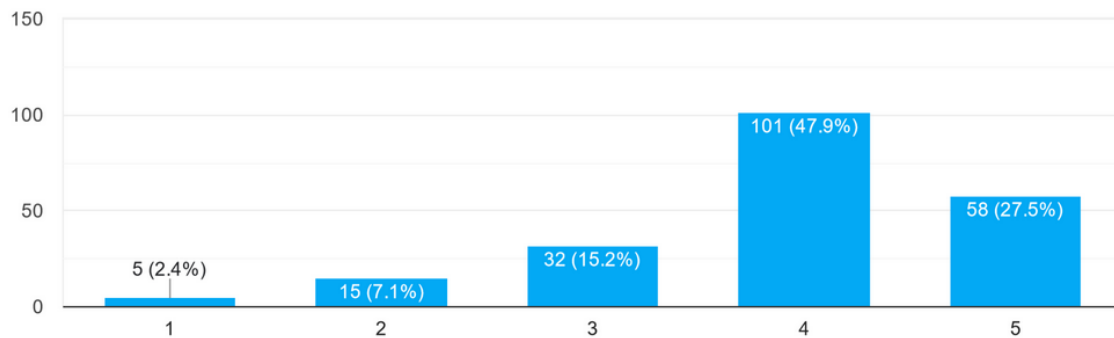
## A.2 Section 2

In this section, each question presented consists of a statement, and the user must answer with how much they agree with it, ranging from "Completely disagree" (1) to "Completely agree" (5).

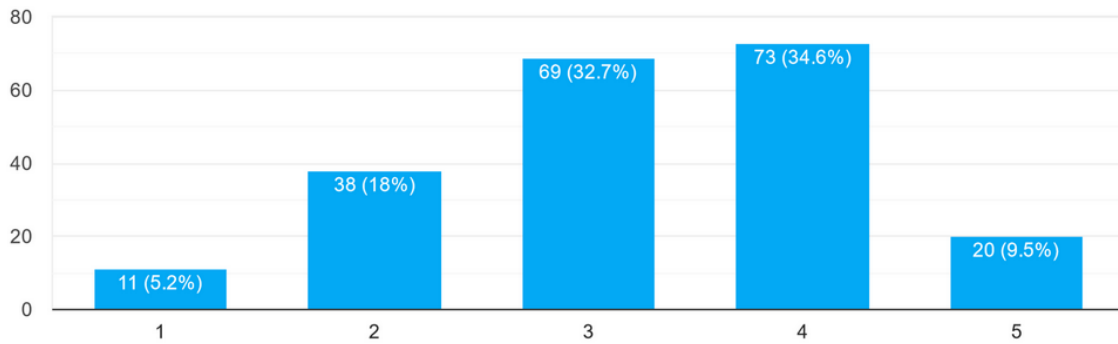
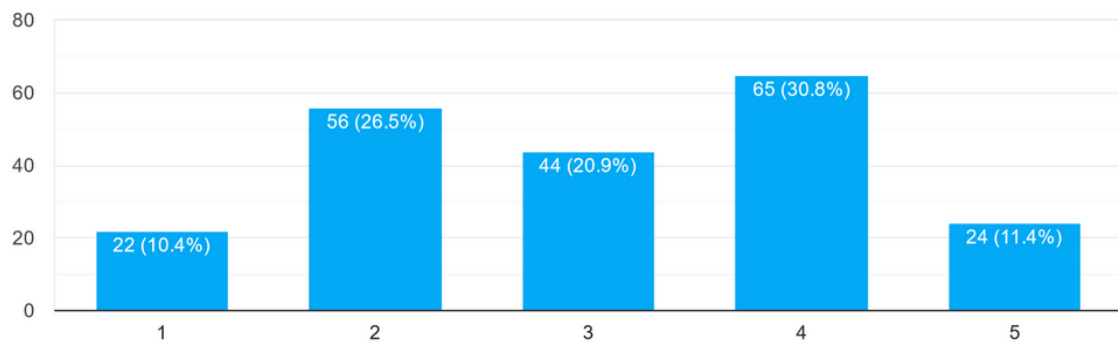
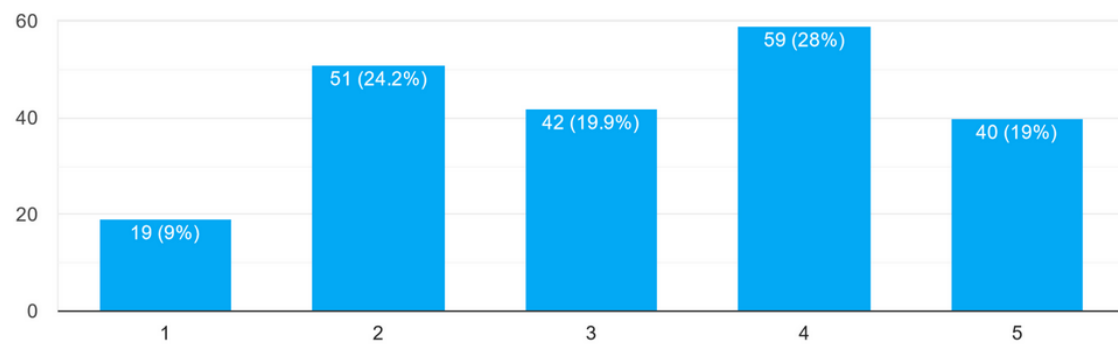
1. This representation (timeline) allows the understanding of an event over time.



2. It's useful to be able see the tweets used to generate the narrative of an event.





**3. The knowledge graph complements the news article.****4. In a situation where time is short, the knowledge graph can replace the news article.****5. The graph eases/simplifies the understanding of a narrative compared to a large set of tweets.**





2. Indicate which of these actors/relations you find interesting, that aren't mentioned in the summary.

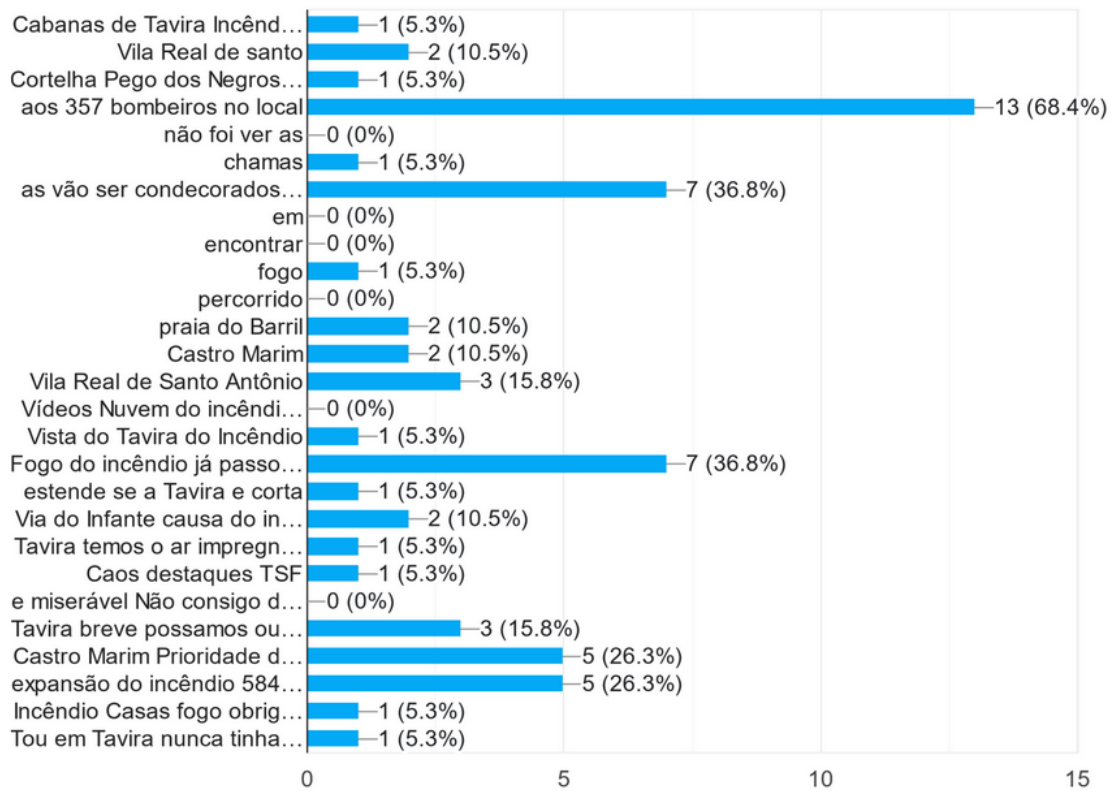


Figure B.2: Interesting actors/relations not present in the summary

3. From the new actors/relations in this time window, which ones do you consider relevant?

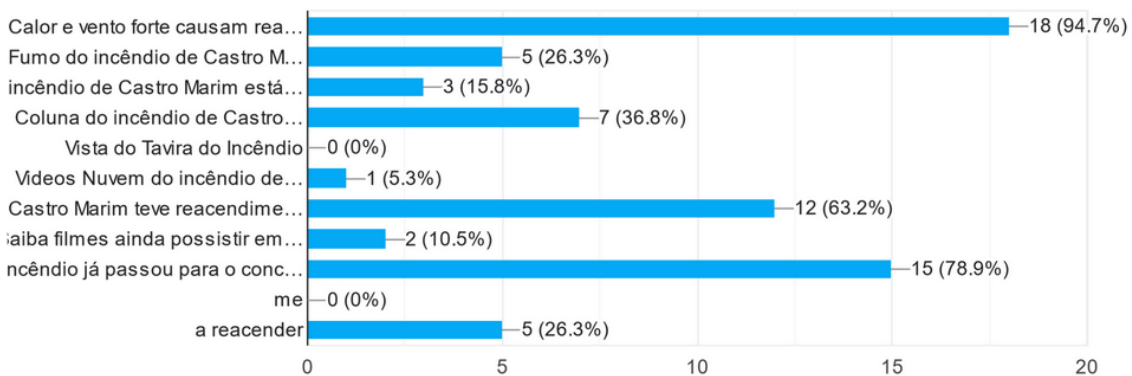


Figure B.3: Relevance of actors/relations new actors in a global time window



2. Indicate which of these actors/relations you find interesting, that aren't mentioned in the summary.

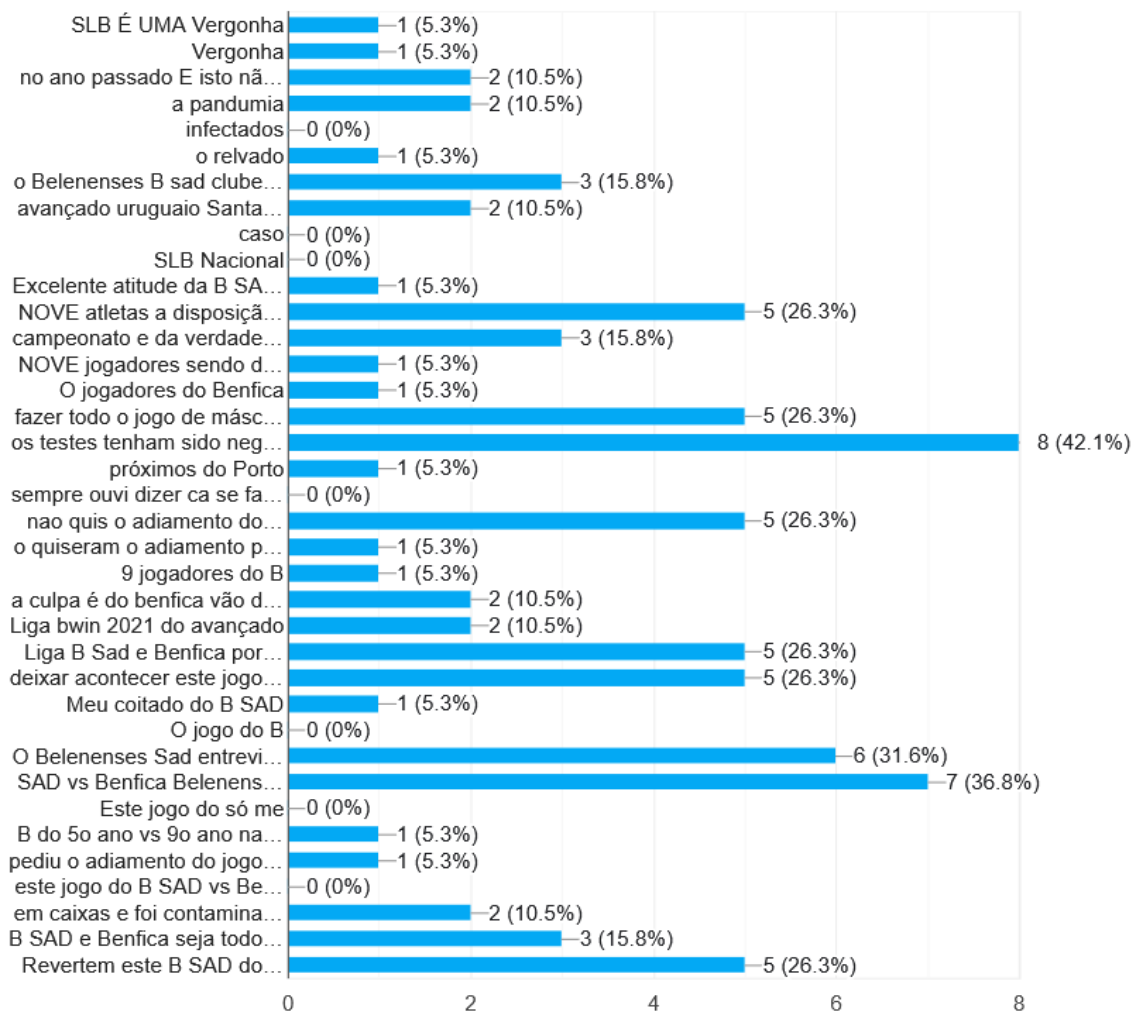


Figure B.5: Interesting actors/relations not present in the summary

**3. From the new actors/relations in this time window, which ones do you consider relevant?**

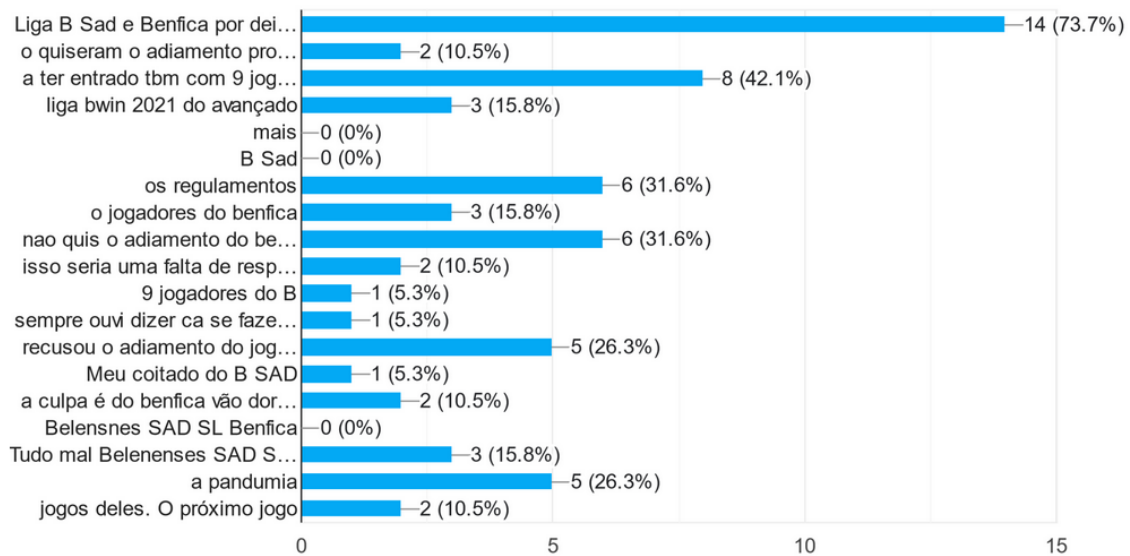


Figure B.6: Relevance of actors/relations new actors in a global time window

### B.3 Section 3 - Shooting in Denmark Mall

1. For each of these actors/relations, indicate the journalistic elements (who, what, when, where?)

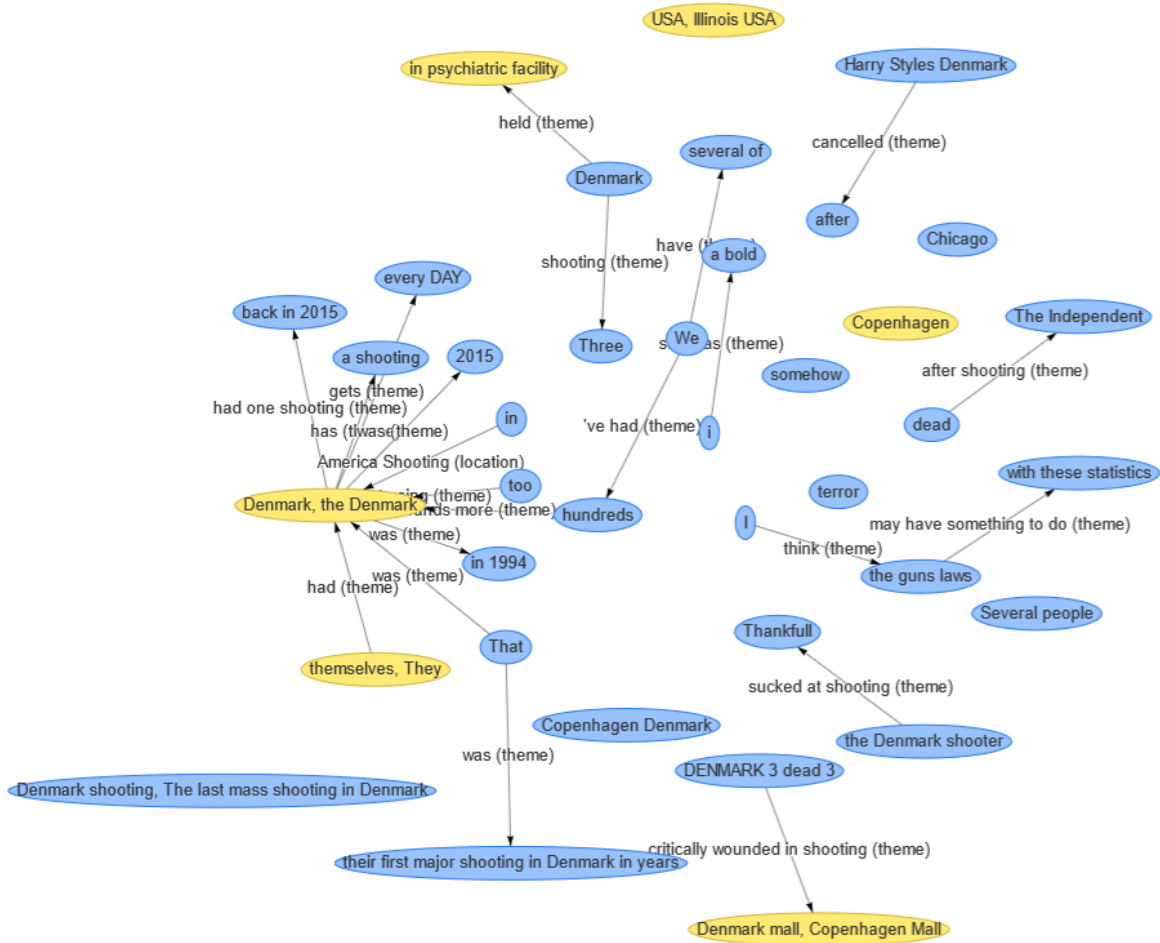


Figure B.7: Knowledge Graph for the event "Shooting in Denmark Mall"



2. Indicate which of these actors/relations you find interesting, that aren't mentioned in the summary.

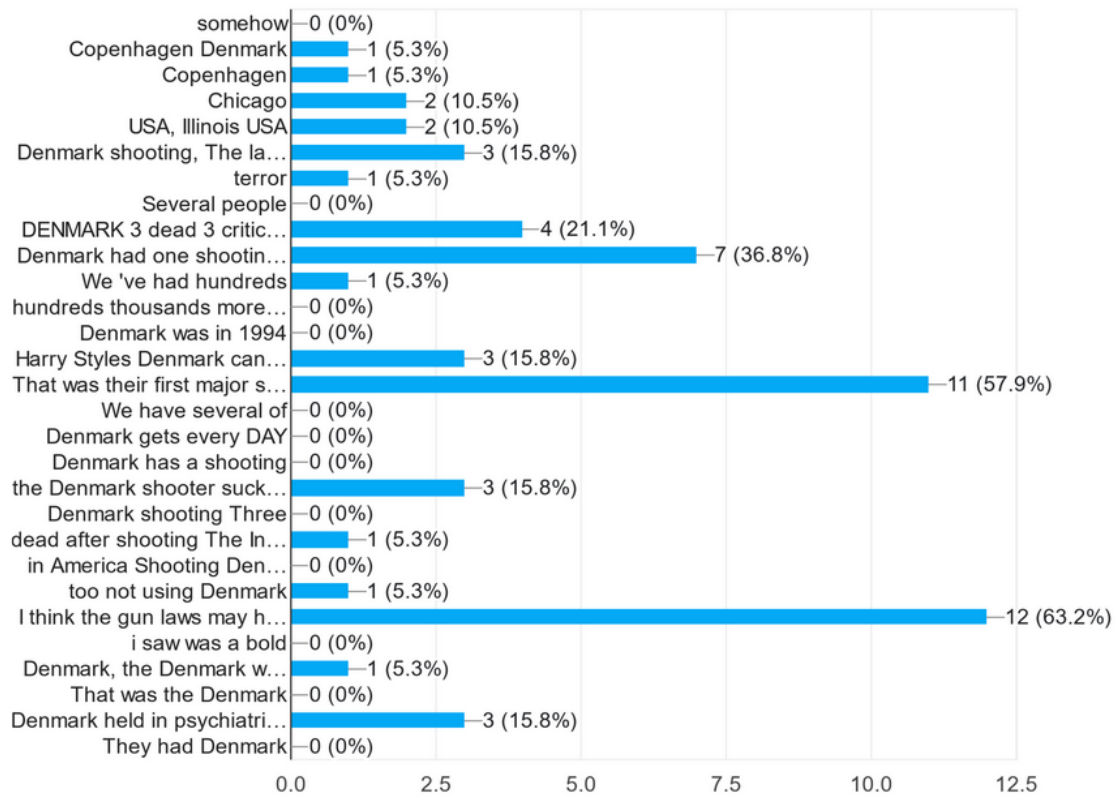


Figure B.8: Interesting actors/relations not present in the summary

3. From the new actors/relations in this time window, which ones do you consider relevant?

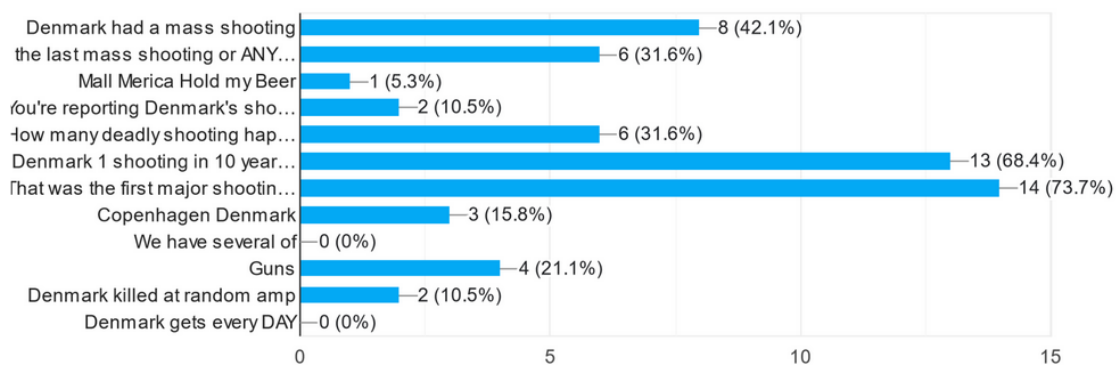


Figure B.9: Relevance of actors/relations new actors in a global time window



# Bibliography

- [1] S. Dixon. [Number of monetizable daily active international twitter users \(mdau\) from 1st quarter 2017 to 2nd quarter 2022](#). [Online], August 2022. Last accessed 28 September 2022.
- [2] MuckRack. [The state of journalism 2022](#). [Online], May 2022. Last accessed 31 August 2022.
- [3] Mark Jurkowitz and Jeffrey Gottfried. [Twitter is the go-to social media site for u.s. journalists, but not for the public](#). [Online], June 2022. Last accessed 31 August 2022.
- [4] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. [Extractive summarization as text matching](#). *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, July 2020.
- [5] Yang Liu and Mirella Lapata. [Text summarization with pretrained encoders](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3728–3738. Association for Computational Linguistics, 2019.
- [6] Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. [Discourse-aware neural extractive model for text summarization](#). *CoRR*, abs/1910.14142, 2019.
- [7] Yixin Liu and Pengfei Liu. [Simcls: A simple framework for contrastive learning of abstractive summarization](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 1065–1072. Association for Computational Linguistics, 2021.
- [8] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. [BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics, 2020.
- [9] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy,

- Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. [Roberta: A robustly optimized BERT pretraining approach](#). *CoRR*, abs/1907.11692, 2019.
- [10] Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. [Gsum: A general framework for guided neural abstractive summarization](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4830–4842. Association for Computational Linguistics, 2021.
- [11] Nazanin Firoozeh, Adeline Nazarenko, Fabrice Alizon, and Béatrice Daille. [Keyword extraction: Issues and methods](#). *Nat. Lang. Eng.*, 26(3):259–291, 2020.
- [12] Eirini Papagiannopoulou and Grigorios Tsoumakas. [A review of keyphrase extraction](#). *WIREs Data Mining Knowl. Discov.*, 10(2), 2020.
- [13] Anand Rajaraman and Jeffrey David Ullman. *Data Mining*, page 1–17. Cambridge University Press, 2011. doi:10.1017/CBO9781139058452.002.
- [14] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. [Yake! keyword extraction from single documents using multiple local features](#). *Information Sciences Journal*, 509:257–289, 2020.
- [15] Rada Mihalcea and Paul Tarau. [Textrank: Bringing order into text](#). In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 404–411. ACL, 2004.
- [16] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. [The pagerank citation ranking: Bringing order to the web](#). Technical Report 1999-66, Stanford InfoLab, November 1999.
- [17] Calvin R. Rensch. [Calculating lexical similarity](#). SIL International Publications in Linguistics. Summer Institute of Linguistics and the University of Texas at Arlington, 1992.
- [18] Sébastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. [Semantic Similarity from Natural Language and Ontology Analysis](#). Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2015.
- [19] Paul Jaccard. [The distribution of the flora in the alpine zone](#). *New Phytologist*, 11(2):37–50, 1912.
- [20] Dan Jurafsky and James H. Martin. [Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition](#). Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, 2009.

- [21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. [Efficient estimation of word representations in vector space](#). In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.
- [22] Nils Reimers and Iryna Gurevych. [Sentence-bert: Sentence embeddings using siamese bert-networks](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.
- [23] Quoc V. Le and Tomas Mikolov. [Distributed representations of sentences and documents](#). In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org, 2014.
- [24] Vasco Campos, Ricardo Campos, Pedro Mota, and Alıpio Jorge. [Tweet2story: A web app to extract narratives from twitter](#). In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II*, page 270–275, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN: 978-3-030-99738-0.
- [25] Purificao Silvano, Antonio Leal, Ines Cantante, Fatima Oliveira, and Alıpio Mario Jorge. [Developing a multilayer semantic annotation scheme based on ISO standards for the visualization of a newswire corpus](#). In *Proceedings of the 17th Joint ACL - ISO Workshop on Interoperable Semantic Annotation*, pages 1–13, Groningen, The Netherlands (online), jun 2021. Association for Computational Linguistics.
- [26] David Sayce. [The number of tweets per day in 2022](#). [Online], August 2022. Last accessed 25 September 2022.
- [27] Quanzhi Li and Qiong Zhang. [Twitter event summarization by exploiting semantic terms and graph network](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(17): 15347–15354, May 2021.
- [28] Koustav Rudra, Pawan Goyal, Niloy Ganguly, Muhammad Imran, and Prasenjit Mitra. [Summarizing situational tweets in crisis scenarios: An extractive-abstractive approach](#). *IEEE Transactions on Computational Social Systems*, 6(5):981–993, September 2019.
- [29] Zhenhua Wang, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. [On summarization and timeline generation for evolutionary tweet streams](#). *IEEE Transactions on Knowledge and Data Engineering*, 27(5):1301–1315, August 2015.
- [30] Nasser Alsaedi, Pete Burnap, and Omer Rana. [Automatic summarization of real world events using twitter](#). *Proceedings of the International AAAI Conference on Web and Social Media*, 10(1):511–514, August 2021.
- [31] Abdelhamid Chellal and Mohand Boughanem. [Optimization framework model for retrospective tweet summarization](#). In Hisham M. Haddad, Roger L. Wainwright, and

- Richard Chbeir, editors, *Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC 2018, Pau, France, April 09-13, 2018*, pages 704–711. ACM, 2018.
- [32] Koustav Rudra, Subham Ghosh, Niloy Ganguly, Pawan Goyal, and Saptarshi Ghosh. [Extracting situational information from microblogs during disaster events: a classification-summarization approach](#). In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 583–592. ACM, 2015.
- [33] Rishab Singla, Sandip Modha, Prasenjit Majumder, and Chintak Mandalia. [Information extraction from microblog for disaster related event](#). In *Proceedings of the First International Workshop on Exploitation of Social Media for Emergency Relief and Preparedness co-located with European Conference on Information Retrieval, SMERP@ECIR 2017, Aberdeen, UK, April 9, 2017*, volume 1832 of *CEUR Workshop Proceedings*, pages 85–92. CEUR-WS.org, 2017.
- [34] Stephen Robertson and Hugo Zaragoza. [The probabilistic relevance framework: Bm25 and beyond](#). *Foundations and Trends in Information Retrieval*, 3:333–389, 01 2009.
- [35] Abdelhamid Chellal, Mohand Boughanem, and Bernard Dousset. [Word similarity based model for tweet stream prospective notification](#). *ECIR 2017 39th European Conference on Information Retrieval*, pages 655–661, Apr 2017.
- [36] Ke Tao, Fabian Abel, Claudia Hauff, and Geert-Jan Houben. [What makes a tweet relevant for a topic?](#) In *Proceedings of the WWW’12 Workshop on ‘Making Sense of Microposts’, Lyon, France, April 16, 2012*, volume 838 of *CEUR Workshop Proceedings*, pages 49–56. CEUR-WS.org, 2012.
- [37] Yajuan Duan, Zhimin Chen, Furu Wei, Ming Zhou, and Heung-Yeung Shum. [Twitter topic summarization by ranking tweets using social influence and content quality](#). In *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*, pages 763–780. Indian Institute of Technology Bombay, 2012.
- [38] Tiago de Melo and Carlos M.S. Figueiredo. [A first public dataset from brazilian twitter and news on covid-19 in portuguese](#). *Data in Brief*, 32:106179, 2020.
- [39] Paulo Rodrigo Cavalin, Flavio V. D. de Figueiredo, Maíra Gatti de Bayser, Luis Gregorio Moyano, Heloisa Candello, Ana Paula Appel, and Renan Souza. [Building a question-answering corpus using social media and news articles](#). In *Computational Processing of the Portuguese Language - 12th International Conference, PROPOR 2016, Tomar, Portugal, July 13-15, 2016, Proceedings*, volume 9727 of *Lecture Notes in Computer Science*, pages 353–358. Springer, 2016.
- [40] Silvia M. W. Moraes, André L. L. Santos, Matheus Redecker, Rackel M. Machado, and Felipe Rech Meneguzzi. [Comparing approaches to subjectivity classification: A study](#)

- on portuguese tweets. In *Computational Processing of the Portuguese Language - 12th International Conference, PROPOR 2016, Tomar, Portugal, July 13-15, 2016, Proceedings*, volume 9727 of *Lecture Notes in Computer Science*, pages 86–94. Springer, 2016.
- [41] Gaspar Brogueira, Fernando Batista, João Paulo Carvalho, and Helena Moniz. [Expanding a database of portuguese tweets](#). In *3rd Symposium on Languages, Applications and Technologies, SLATE 2014, June 19-20, 2014 - Bragança, Portugal*, volume 38 of *OASICs*, pages 275–282. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014.
- [42] Alexis Dusart, Karen Pinel-Sauvagnat, and Gilles Hubert. [Tssubert: Tweet stream summarization using BERT](#). *CoRR*, abs/2106.08770, 2021.
- [43] Axel Suarez, Dyaa Albakour, David Corney, Miguel Martinez, and José Esquivel. [A data collection for evaluating the retrieval of related tweets to news articles](#). In *Advances in Information Retrieval*, pages 780–786, Cham, 2018. Springer International Publishing.
- [44] David Corney, Dyaa Albakour, Miguel Martinez, and Samir Moussa. [What do a million news articles look like?](#) In *Proceedings of the First International Workshop on Recent Trends in News Information Retrieval co-located with 38th European Conference on Information Retrieval (ECIR 2016), Padua, Italy, March 20, 2016.*, pages 42–47, 2016.
- [45] Ricardo Campos, Arian Pasquali, Adam Jatowt, Vítor Mangaravite, and Alípio Mário Jorge. [Automatic Generation of Timelines for Past-Web Events](#), pages 225–242. Springer International Publishing, Cham, 2021.
- [46] Jiexin Wang, Adam Jatowt, and Masatoshi Yoshikawa. [Archivalqa: A large-scale benchmark dataset for open-domain question answering over historical news collections](#). In Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai, editors, *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 3025–3035. ACM, 2022.
- [47] Lusa. [Incêndio que deflagrou em castro marim dominado](#). [Online], August 2021. Last accessed 25 September 2022.