TÉCNICO LISBOA

Event Identification in STRING

José Paulo de Oliveira Rodrigues Marques Dias

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Doutor Nuno João Neves Mamede
Prof. Doutor Jorge Manuel Evangelista Baptista

Examination Committee

Chairperson: Prof. Doutor Miguel Nuno Dias Alves Pupo Correia
Supervisor: Prof. Doutor Nuno João Neves Mamede
Member of the Committee: Dr. Helena Gorete Silva Moniz

November 2018

Acknowledgements

First I would like to thank my supervisor, Professor Nuno Mamede, for the guidance provided throughout this work, whose experience and help made this work possible.

I would, also, like to thank my co-supervisor, Professor Jorge Baptista, for the insight on several topics and the attention to detail that helped the improvement of this work.

A special thanks to my family, specially my parents and my sister, whose constant support and insistence have enabled me to finish this work.

And a final word of thank you to all my friends that have accompanied me during this journey, specially Anisa Shahidian, António Pacheco and Luis Guerra, that have shared my torment.

Lisboa, December 18, 2018

José Paulo de Oliveira Rodrigues Marques Dias

To my parents and my sister

António, Fátima and Inês

Resumo

Identificação de eventos em textos é uma tarefa importante no Processamento de Língua Natural (PLN), por permitir que ocorra a extração de informação de um modo estruturado, o que pode ter múltiplas aplicações nas actividades de sumarização automática e de reconhecimento de eventos.

O trabalho descrito neste documento foca-se na expansão e melhoramento da tarefa de identificação de eventos na cadeia de processamento STRING (Statistical and Rule-Based Natural Language Processing), desenvolvida no Laboratório de de Sistemas de Língua Falada (L2F) do Instituto de Engenharia de Sistemas e Computadores - Investigação e Desenvolvimento em Lisboa (INESC-ID).

Este documento apresenta as descrições de sete tipos de eventos que foram desenvolvidos, Crime, Trial, Prision, Location Static, Location Visit, Public e Ephemerid, e os critérios que determinam quando é que uma situação pode ser considerada como sendo um evento. Adicionalmente, o documento descreve, para cada tipo de evento, as relações que podem ser extraídas e que revelam informação relevante acerca do tipo de evento em que se enquadram.

O sistema STRING foi expandido de modo a acomodar o novo tipo de eventos e alguns dos tipos de eventos já existentes foram melhorados de modo a providenciar um melhor desempenho na tarefa de identificação de eventos.

Um conjunto de 350 frases, extraidas do corpus não anotado dos registos do Parlamento Português, foram usados durante a avaliação da implementação. O conjunto foi dividido em conjuntos de 50 frases para cada avaliação de evento. Cada conjunto foi executado pelo sistema modificado e os resultados foram avaliados de acordo com a capacidade do sistema de identificar novos eventos. A avaliação da identificação de eventos resultou uma medida-f total de 51%, com as medidas-f individuais, de cada evento, de 65% para Crime, 82% para Trial, 62% para Prision, 55% para Static, 11% para Visit, 14% para Public e 66% para Ephemerid. A avaliação da identificação de relações resultou numa medida-f total de 74%, com as medidas-f individuais de 65% para Crime, 89% para Trial, 46% para Prision, 59% para Static e 62% para Ephemerid. Os resultados para os eventos de Visit e Public tiveram ambos 100%, mas tiveram um numero extremamente reduzido de relações para avaliar.

Abstract

Event identification in texts is an important task in Natural Language Processing (NLP), as it allows for the extraction of information in a structured way, which can have multiple applications in automatic summarization an event reconnaissance.

The work described in this document is focused on the expansion and improvement of the task of event identification in the processing chain STRING (Statistical and Rule-Based Natural Language Processing), developed at the Laboratory for Spoken Language Systems (L2F) of the Institute of Systems Engineering and Computers Research and Development in Lisbon (INESC-ID).

This document presents the descriptions of seven types of events that were developed, i.e. Crime, Trial, Prision, Location Static, Location Visit, Public and Ephemerid, and the criteria that determines when a situation can be considered an event. Additionally, the document presents, for each event type, the relations that can be extracted and that reveal relevant information about the event they belong to.

The STRING system was expanded in order to accommodate the new types of events and some of the events that already existed were further developed in order to provide a better performance on the task of event identification.

A set of 350 sentences, extracted from a non-annotated corpus of the recordings of the Portuguese Parliament, were used for the evaluation of the implementation. The set was divided so that 50 sentences were used per event evaluation. Each set was ran through the modified system and the results were evaluated based on the systems ability of identifying the new events and the event relations. The evaluation of event identification yielded a total f-measure of 51%, with the individual f-measures, of each event, of 65% for Crime, 82% for Trial, 62% for Prision, 55% for Static, 11% for Visit, 14% for Public and 66% for Ephemerid. The evaluation of relation identification yielded an overall f-measure of 74%, with the individual f-measures, of each event, of 65% for Crime, 89% for Trial, 46% for Prision, 59% for Static and 62% for Ephemerid. The Visit and Public results for relation identification yielded both 100%, but had an extremely reduced number of relations to evaluate.

Palavras Chave
Keywords

Palavras Chave

Processamento de Língua Natural (PLN)

Identificação de Eventos

Identificação baseada em regras

Português

Keywords

Natural Language Processing (NLP)

Event Identification

Rule-based identification

Portuguese

Contents

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence

MARv Morphosyntactic Ambiguity Resolver

NER Named Entity Recognition

NLP Natural Language Processing

POS Part of Speech

RuDriCo Rule-Driven Converter

STRING Statistical and Rule-based natural language processing

XIP Xerox Incremental Parser

XML Extensible Markup Language

ViPEr Verb for European Portuguese

1 Introduction

1.1 Motivation

Ever since the creation of writing, human being have been able to encode information in the texts they write. These texts provide a way to share knowledge among people, taking into account that they have the ability to properly understand the information that is present in the texts.

Proper understanding of a written text is not a trivial matter, as it requires the reader to know all the rules of the language at hand. But even this may not be enough if the intent of the writer is not properly conveyed in the words he has written. Therefore, the analysis texts cannot be considered a trivial matter. It is necessary to determine the both the explicit information and the implicit information.

Nowadays there are methods to automatically extract information (Hogenboom et al. 2011) from texts. These methods are non-trivial as there is considerable difficulty in making consistent connection between the words that compose a text and ensure that the information that was gathered is consistent with what was actually written and what can be taken from it, when analyzed by a person. The extraction of useful information proves yet another challenge when taken into account that, assuming that an automatic method as processed through all the inherent rules of a language, it is necessary to deal with the ambiguity of human language, which can prove quite challenging and very difficult for a machine to deal with.

Natural Language Processing (NLP) is one area of Artificial Intelligence (AI) that specializes in the automatic comprehension of natural languages (human language). Named Entity Recognition (NER) is one of the tasks of NLP that deals with the identification, classification and localization of Named Entities (NE), which can be any real world object with physical existence, e.g, the person "Pedro", or be completely abstract, e.g. the month "January". NE are usually denoted by proper nouns and can be divided into several categories, such as the names of persons, organizations, locations, temporal and numeric expressions, etc.

The STatistical and Rule-based Natural lanGuage processing chain (STRING) (Mamede et al. 2012) is a text processing chain developed at L2F (Spoken Language System Lab), that has the capability of processing very large corpora and is able to perform several NLP tasks such as NER (Oliveira 2010) (da Silva Romão 2007) (Loureiro 2007), extraction of Relations between entities (de Almeida Santos

2010), Anaphora Resolution (Marques 2013), and others. This system is further explained in Section 2.2.

This work focuses on the task of event identification in STRING which is performed by its parsing module, using Xerox Incremental Parser (XIP) (Ait-Mokhtar et al. 2002), which is further explored in Section 2.2. The identification of events consists in the identification of the predicative elements of sentences and the attribution of semantic roles to the arguments and circumstantial elements of those semantic predicates. This is a generic procedure, that is not considered in this work.

The definition of event can be somewhat vague and, depending of different sources, acquire slightly different meanings but, it can be tentatively defined as a linguistic expression that denotes a process or a state, under a specific set of circumstances, and that involves a certain number of entities.

There are different typologies of events in literature (Baptista et al. 2011), for example, distinctions may be made between "man-made" (e.g. meeting, party, etc.) and "natural" (e.g. rain, earthquake, etc.) events. In STRING an event is formally represented using the following notation:

```
EVENT_LEX(<word>,<subtype>)

EVENT_<EVENT_TYPE>_<features>(<word>)

EVENT_<RELATION_TYPE>_<features>(<word>,<arg1>)
```

- `EVENT_LEX(<word>,<subtype>)` is a dependency that associates the node `<word>` that represents the event to a specific event `<subtype>`.
- `EVENT_<EVENT_TYPE>_<features>(<word>)` associates the node `<word>` with an `<event type>`, which will define the types of relations that can be extracted from the sentence. Relations are dependencies that link the event node word to the named entity that is involved in the event.
- `EVENT_<RELATION_TYPE>_<features>(<word>,<arg1>)` is a dependency that links the node `<word>` with the arguments `<arg1>`, that are involved with the event, and associates them to the event relation that is extracted for the event in question.

Some dependencies may also have optional `<features>` that give further information on the context of that dependency, e.g. the identification of a person gender through the features `_2M`, for masculine, and `_2F`, for feminine.

STRING is able to identify a variety of distinct event types that can be associated with different dependency subsets, such as *Lifetime*, *Business*, *Crime*, *Location*, *Ephemerids* and *Public* events. Each of these types (and associated subsets of dependencies) deal with certain aspects of events that share similar context, e.g. the Lifetime event dependency is associated with the events that describe the life of an individual. For this reason, there are events that define a person's *birth* (nascimento) (e.g. "A Maria

nasceu hoje." ("Maria was born today.")), *death* (morte) (e.g. "O avô do João morreu à dois dias." ("João's grandfather died two days ago.")) and *age* (idade) (e.g. "O António tem dez anos." ("António is ten years old.")), as well as their *family relations* (família) (e.g. "O Rui é irmão do André." ("Rui is André's brother.")), *marital status* (casamento) (e.g. "O Bernardo e a Maria casaram." ("Bernardo and Maria got married.")) and *education* (educação) (e.g. "O José terminou o seu curso em Informática em 2016 no IST." ("José finnished his degree in Informatics in 2016 in IST.")).

Events may be divided into different subtypes that give further context inside the major event type, i.e. the `crime` event type can be divided into several subtypes, e.g. `homicídio` (homicide), `roubo` (robery), `corrupção` (corruption), etc., that help classify the event. Some event relations may also be unique to specific event subtypes, e.g. the `sentence` relation can only be extracted for the `prision` event when its subtype is `prisão` (prison).

1.2 Goals

The goals of this work involve the improvement and expansion of the current functionality of the STRING chain for the identification and classification of events.

The system previously identified a small set of crime related events. This set was expanded during this work, in order to identify additional crime events and improved, in order to better identify the current existing sets.

Additionally, the system was expanded in order to be capable of identifying the events of location related events, public events and ephemerids.

Crimes can be divided into three subtypes:

- the `Crime` subtype, which associates a person to a crime he/she has committed (e.g. "Um homem roubou o ouro a um casal de idosos." ("A man stole the gold from an elderly couple."));
- the `Trial` subtype, that associates a person to his/her trial, his/her conviction or absolution for a crime, (e.g. "O Pedro foi condenado a um ano de prisão por crimes de corrupção." ("Pedro was sentenced to one year of jail for crimes of corruption."));
- the `Prision` subtype, that associates a person to his/her arrest or imprisonment for a crime (e.g. "O Pedro foi preso pelo homicídio de 2 pessoas." ("Pedro was arrested for the murder of two people.")).

Both the examples sugested for `Trial` and `Prision` also represent examples of `Crime` events. The reasoning behind this will be explained further in 3.

Considering, the previously referred example of a Crime event:

"Um homem roubou o ouro a um casal de idosos." ("A man stole the gold from an elderly couple.")

in this example, the *culprit* "Um homem" ("A man") is associated with the *crime* of "roubar" ("steal").

The representation of the event dependency for this sentence, and its associated relations, are as following:

- Um homem roubou ouro a um casal de idosos.

```
EVENT_LEX(roubou,roubo)
EVENT_CRIME(roubo)
EVENT_AGENT(roubou,homem)
EVENT_VICTIM(roubou,de idosos)
EVENT_OBJECT(roubou,ouro)
```

- `EVENT_LEX(roubou, roubo)` associates the word "roubou" to the predefined subtype of `roubo` that is one of subtypes that exist for this event;
- `EVENT_CRIME(roubo)` associates the word with the major event type `Crime`;
- `EVENT_AGENT(roubou, homem)` associates the word "homem" with the entity that is responsible for the event;
- `EVENT_VICTIM(roubou, de idosos)` associates the words "de idosos" as having the role of victim in the event;
- `EVENT_OBJECT(roubou, ouro)` defines the object that is being stolen.

Location-related events (Locative events) associate entities to a certain location. There are two subtypes of location events:

- the `Local-stat` subtype, that associates any sort of entity to a fixed location (e.g. "A Torre de Belém situa-se em Lisboa." ("Torre de Belém is located in Lisbon."));
- and the `Local-visit` subtype, that associates a human entity to a location or a building when it is implied that the entity had to move to get to that location or "visited" that location (e.g. "O José foi a Espanha." ("José went to Spain.")).

Consider the previously referred example for `Local-stat`:

"A Torre de Belém situa-se em Lisboa." ("Torre de Belém is located in Lisbon.")

in this sentence, "Torre de Belém" is associated with the city of Lisbon. Note that this kind of relation association can also be done for a person, as in, "O Pedro está em Lisboa." ("Pedro is in Lisbon."). The formal representation for this event should be as following:

- A Torre de Belém situa-se em Lisboa.

```
EVENT_LEX(situa-se,local-stat)
EVENT_LOCATIVE(situa-se)
EVENT_OBJ(situa-se,Torre de Belém)
EVENT_PLACE(situa-se,Lisboa)
```

Public events relate to any sort of event where there is a gathering of people, such as a *party* (festa) or a *meeting* (reunião) and it associates all the participants of the event to the location of the event and to when it occurs (e.g. "O João foi à festa no Porto." ("João went to the party in Porto.")).

Ephemeride events celebrate the anniversary of a particular relevant event. They are divided into three specific subtypes, nascimento (birth) (e.g. "Celebram-se os 100 anos do nascimento de Nelson Mandela." ("We celebrate the 100 years of the birth of Nelson Mandela.")), morte (death) (e.g. "Celebram-se os 9 anos da morte de Michael Jackson." ("We celebrate the 9 years of the death of Michael Jackson.")) and efémeride (ephemeride) (e.g. "Celebram-se os 100 anos da abertura do Instituto Superior Técnico." ("We celebrate the 100 years of the inauguration of Instituto Superior Técnico.")). This last subtype includes all other celebratory events outside the ones related to the birth or death of a person, i.e. important national dates, the anniversary of the opening of an institution, etc.

The dependency representation for the last given example should be as follows:

- Celebram-se os 100 anos da abertura do Instituto Superior Técnico.

```
EVENT_LEX(abertura,efeméride)
EVENT_EPHEMERIDE(abertura)
EVENT_EVENT_ARG(abertura,Instituto Superior Técnico)
EVENT_DURATION(abertura,100 anos)
```

1.3 Document Structure

This document is structured as follows:

Chapter 2 will begin with a presentation and description of four experiments that used different methods to solve the issues of identification and classification of real world events and, following that, a description of the architecture of STRING is presented.

Chapter 3 presents the basis for the identification of the different event types that were implemented throughout this work.

Chapter 4 details the process of implementation and provides insight for the decisions taken throughout it.

Chapter 5 describes the methods used to evaluate the implementation, present a discussion of the results obtained and provide the conclusions that can be taken from them.

At last, Chapter 6 provides an overview of the entire work and will point out any future work that can be done in order to further improve the system.

2 State of the Art

This chapter begins with a description of few experiments that involve the identification and classification of real world events.

The first of these experiments uses a rule-based approach to identify and classify events and the following experiments use distinct automatic approaches to achieve the same objectives. These approaches use methods such as: Support Vector Machines, Local Sensitive Hashing and a combination of several classification methods in order to classify events.

Then follows a description of the architecture of the STRING chain, which is the system that was expanded throughout this work.

The chapter is concluded with a small recap of the different experiments and of STRING.

2.1 Related Work

2.1.1 Knowledge-based Approach for Event Extraction from Arabic Tweets

Social media websites such as Twitter are capable of providing the most up-to-date news about a certain topic, even before the topic is covered by actual news dedicated media. Extraction of information from Tweets is troublesome because of the limitation imposed on the tweets themselves, as they are of limited size and self-contained, lack important contextual information, and usually are of personal nature that mostly describe the authors' activities.

(Al-smadi and Qawasmeh 2016) use a Knowledge-based approach in order to extract events from Arabic Tweets. They use an unsupervised rule-based event extraction technique, which also provides named entity disambiguation for some event-related entities.

The extracted events and their related entities are populated onto the event knowledge base, where tagged tweet entities are linked to their entity representation in the knowledge base.

The event extraction technique extracts named entities associated with events and specific event arguments. These arguments may identify the event agent, location, target (i.e where the event is taking

place in the location), event trigger (i.e. the linguistic expression that refers to the tweet), product (i.e. when a product is being announced) and the event time (i.e. when it is taking place).

The Twitter Streaming API (twi) was used to collect the data by searching the tweets for temporal expressions, which resulted in a total of 3000 tweets. These tweets were then evaluated using an unsupervised event tagging approach that resulted in a subset of 1000 tweets, tagged as either instant type events or interval type events. The output of the annotation was then manually evaluated, resulting in a final dataset with a total of 122 interval type events and 878 instant type events.

Text preprocessing was done using the AraNLP Java-based package (Althobaiti et al. 2014). The texts were first cleaned of non Arabic words, hyper-links, hash-tags, punctuations, Arabic diacritics and other symbols, and then they were normalized by replacing some Arabic characters with simplified equivalents. The tweets were POS tagged with the AraNLP (Althobaiti et al. 2014) POS tagging tool and the named entities in the tweets were automatically annotated using an hybrid approach of machine learning and linked data.

Extracted events are represented using an ontological knowledge base that adopts linked data and was extended to OWL-Time Ontology. This representation allows the linking of entities with their events and allows the representation of events as instant or interval.

The extraction of events is done with the use of the following processes: extraction of event mentions, named entity recognition and disambiguation, and temporal resolution.

A rule-based approach based of the Arabic Annotation Guidelines for Events (Consortium 2008), provided by the Linguistic Data Consortium (LDC), was used in order to proceed with the extraction of event mentions, consisting of the event triggers and event time arguments of the tweet. The POS tags of the tweets were used by the syntactic rules to extract the event triggers and the event time.

The rules in Table 2.1 were applied depending on the type of POS tag present in the tweet. One can interpret the first entry of the table as such: if a verb (VB) is present in the tweet then one of the rules VB + NN (i.e. verb followed by noun), VB + VB (i.e. verb followed by verb) or VB (i.e. verb only) is applicable depending on the following word.

The rules associated with verbs and nouns were used to extract the event triggers while the ones associated with the cardinal number were used to extract the event time.

The system verifies the time expressions present on the tweet in order to determine the type of the event (i.e. interval or instant).

In the step of named entity recognition and disambiguation, any tweet that was previously identified to have an event, has the event agent and event location arguments tagged. Disambiguation is then applied, as it is necessary to differentiate between location, organization and person.

Tweet POS Tag	Trigger Rules
Verb (VB)/ Verb phrase (VBP)	VB+NN
	VB+VB
	VB only
Noun (NN) / Noun phrase(NNP)	NN+NN
	NN/NNP Only
	NN/NNP + VB/VBP
Cardinal number (CD)	CD + DTNN (Determiner)
	CD + DTNN +CD

Table 2.1: Trigger rules

Task	Event Type	
	Instant	Interval
T1: Event Trigger Extraction	75.5%	78.6%
T2: Event Time Extraction	86.7%	92.6%
T3: Event Type Identification	97.9%	95.9%

Table 2.2: Task accuracy based on event type

The temporal resolution process is then applied in order to determine when the event, presented in the tweet, is taking place. This is necessary as it is possible to represent an event date in several different ways (e.g. “Next monday.” and “The first day of the month.” may represent the same day in the calendar.).

The corpus of 1000 tweets was used to evaluate the three tasks of event trigger extraction (T1); event time extraction (T2); and event type identification (T3). To measure the accuracy of the tasks, the number number of correctly predicted values are divided by the number of overall tweets. To note that out of the 1000 event tweets, 122 were of type interval and 878 were of type instant.

The accuracy testing results for each of the tasks when applied to the different event types can be seen in Table 2.2.

Overall, the Event type identification returned the greatest accuracy. The performance of the first two tasks was influenced by existence of POS tagging errors from incorrect tag assignment and conflict between rules. These errors were attributed to the structure of the POS tagger used, as it was not the most suitable.

2.1.2 Word Level Event Identification

The Document Understanding Conference (DUC) shared tasks (Dang 2006) attempted to apply multi-document summarization to newswire and newspaper documents sets that describe the same series of events using different sources in order to produce a proper 250 word summary. Because of the nature

of the majority of the news in the articles (historical and biographical information), the summaries are structured as a time-line of events, i.e. they exhibit the information in an ordered way.

(March and Baldwin 2008) detail the studies that were done in order to produce an Event-Based Summary system that was later turned into a component of the DUC 07 summarization system of (Stokes et al. 2007). The original system used sentence level classification and was effective at answering questions centered on specified topics (e.g. "What countries have the most health problems?"), but had difficulties at answering questions that required the listing of events (e.g. "What are the most significant events of the life of the actor Ian McKellen?"). Support Vector Machines (SVM) are used in this study to find the most effective representation for the event references in the given newswire and newspaper text documents.

In the new system component, a question is classified as either being an event-based question or a general question, and the corresponding summary is generated by the original system, if the question is a general question, or by a dedicated event-based system, if it is an event-based question.

To create the event-based time-line, three main tasks had to be performed:

1. the identification and grouping of event references;
2. the association of a time stamp to each identified event;
3. the identification of prominent events that must be included in the final summary.

This work focuses, solely, on the identification of the event references and, for this, (March and Baldwin 2008) opted to do the identification at the word level instead of at sentence level. In the sentence level, the identification of events is mostly done by finding temporal cues that indicate the occurrence of an event. It is also possible to detect the occurrence of events without temporal cues. For example, the appearance of the word "*earthquake*" in a newspaper can indicate the occurrence of an earthquake. This is not always true as the word may appear with an informative context which does not relate to the occurrence of a specific event.

The experiment was carried out using the TimeBank 1.1 corpus, which consists of 186 news articles, extracted from the Wall Street Journal and the Associate Press newswire. The corpus was marked using the TimeML specification language standard 1.1. (Pustejovsky et al. 2005) The articles contained EVENT tags, which are an event-time structure proper to TimeML and that denote events at both sentence and word level. These EVENT tags contain an unique id, a tense, an aspect, a class and, possibly, another event to which the current event may be associated. The authors disregarded the classes of events for this experiment and instead chose to use a binary classification of events where a word is either identified as an event or a non-event.

The TimeBank data was first preprocessed in order to remove XML tags, remove punctuation and to split the texts into sentences and words. Then the Standford POS tagger was used to assign a POS tag to each word.

The classifications were carried out in a third-party training and evaluation platform named WEKA (Witten et al. 2011) and, after a preliminary testing phase, it was determined that Support Vector Machine (SVM) was the best classification algorithm, when compared with decision trees and Naive Bayes. Using stratified 10-fold cross-validation during the entire process, the text fragments were randomly divided to different combinations of training and testing splits.

Sentence level classification is first used to remove non-event sentences so that only sentences containing events are passed onto the word level classifier. A word feature vector is used to represent the words being classified, where each unique word corresponds to a unique feature, for this the text is transformed into a vector with N entries, where each entry is an unique word in the TimeBank corpus.

Five types of text representation techniques are explored in the experiment: the use of a Context Window (0 to 3 preceding words) to identify the events, the Feature Representation of the context words as a bag-of-words or as a ordered list of word positions (1 - current word, 2 - previous word, 0 - other), the removal of Stop Words, the use of POS tagging instead of word feature for context words and Feature Generalization (i.e. grouping numeric expressions and named entities).

To evaluate the experiment, F-Score (2.3) was used has the primary evaluation metric, which is based on Recall (2.2) and Precision (2.1).

$$Precision = \frac{\#\quad words\quad correctly\quad classified\quad as\quad events}{\#\quad words\quad classified\quad as\quad events} \tag{2.1}$$

$$Recall = \frac{\#\quad words\quad correctly\quad classified\quad as\quad events}{\#\quad words\quad that\quad are\quad events} \tag{2.2}$$

$$F - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.3}$$

The baseline is a unigram tagger where the word is labeled with the most frequent label.

Table 2.3 shows the results of testing different feature combinations. Notice that Feature Generalization was not used in the testing as some of the groupings that were done were causing loss in the system's ability to properly disambiguate. This was due to some wrong groupings of words.

The simple use of Bag-of-Words representation revealed very poor results, when compared with the

Feature Combination	One Word Context	Two Words Context
Bag-Of-Words	0.51	0.46
No Stop Words + Bag-of-Words	0.65	0.56
Ordering	0.68	0.65
No Stop Words + Ordering	0.74	0.68
No Stop Words + Ordering + POS	0.75	0.76

Table 2.3: F-Score for different feature combinations. (Baseline at 0.653 F-Score)

baseline of 0.653 F-Score but, with the removal of stop words, the performance was greatly increased.

Changing representation to a list of ordered words proved yielded far greater results, which were now on par with the baseline. Further combining other features further increased the results of the evaluation.

Overall, increasing the context size resulted in a decrease in performance for every feature combination, with an exception to the use of the ordered list of words with no stop words and with the use of POS to represent each context word, which had an increase of performance.

A final analysis of the results revealed that maintaining word order had the biggest impact on the classification, the usage of POS increased the overall performance of the system and the removal of stop words improved it just very slightly. Due to the small amount of the data available (around 1540 words in the data that occurred as events) it was estimated that the results gained for the higher context words could vastly improve with the increase of the number of training instances.

2.1.3 Streaming First Story Detection with application to Twitter

Extraction of information from continuously growing environments, such as social networks, proves to be a large challenge mostly because of the speed with which new information is created and the difficulty for normal identification systems to keep up with the constant growing set of information. Twitter is one of this environments where new information is constantly appearing, specially when it is related to important news.

(Petrović et al. 2010) address the problem of detecting new events from a stream of Twitter posts by developing a system, based on First Story Detection (FSD), which is one of the subtasks of topic detection and tracking (Allan 2002), that is capable of detecting events on a web-scale corpus through the use of a locality-sensitive hashing algorithm.

In FSD, documents are represented as vectors where the coordinates represent the frequency of a term in the document. A new document is considered to be a first story whenever a it's similarity to the closest document, when compared to all previous documents, is below a certain threshold.

Detection of events from tweets is difficult because of the large volume of data, the large level of noise present in it and because of the speed required to efficiently process such large amount of data. (Petrović et al. 2010) dealt with these issues by developing a FSD system that processes a new document in a constant time, while maintaining a constant space. The constant processing time was achieved with the use of Local Sensitive Hashing (LSH) which dramatically reduces the time needed to find a closest neighbor in the vector space. The constant space was achieved by limiting the amount of documents that can be in memory at any given time.

LSH decreases the dimensionality of high-dimensional data. It hashes each item it receives into buckets in such a way that the probability of collision with similar items increases the more similar the items are. Based of (Allan et al. 2000), they chose to calculate the distance between documents by calculating the cosine between their vectors. The collision probability equation is given in Equation 2.4 and was based of an hashing scheme proposed by (Charikar 2002).

Because this work deals with textual documents, the measure of distance is the cosine between two documents and the probability of two documents colliding is given by the function in Equation 2.4.

$$P_{coll} = 1 - \frac{\theta(x, y)}{\pi} \tag{2.4}$$

where θ(x, y) is the angle between document x and document y.

The buckets are defined by intersecting the space with random hyperplanes. An increase of hyperplanes will decrease the probability of collision with both similar and non-similar documents while increasing the time needed to compute the hash values. To deal with this, multiple hash tables are necessary so that the chance of collision with similar neighbors increases. The number of hash tables (L) is given by the function in Equation 2.5.

$$L = log_{1-P_{coll}^{k}} \delta \tag{2.5}$$

where k is the number of hyperplanes and δ is the desired probability of missing a closest neighbor.

A variance reduction strategy was implemented in order to further support the system, which even with the use of LSH, was still fairly inefficient and presented a lot of variance. This happened because LSH is only able to identify the real closest neighbor if it is close to the query point, failing otherwise. Variance reduction was achieved by searching through the inverted index and only comparing the query with a fixed number of most recent documents. The use of variance reduction in combination with LSH

and the limitations imposed on processing time and item space significantly reduces the running time.

The FSD system assigns a novelty score to each tweet and finds the most similar tweets, which are used to generate threads with a close level of similarity. The smaller the threads the more specific they become. For the experiments, it was decided that the threads should not be too big nor too small. The growth rate of the threads reveals which are the most interesting ones.

The FSD system was used to detect new events from a collection of Twitter data, collected using the Twitter Streaming API between April 2009 and October 2009. The corpus consisted of 163.5 million timestamped tweets with over 2 billion tokens. The tweets contained only ASCII characters and were cleaned of words that had @ and #.

To measure the efficiency of the system, the tweets returned from the system were manually labeled has either Event, Neutral and Spam. The returned events were only the first tweet in a thread, which described the topic of that thread. Spam was defined as being advertisements, automatic weather and radio updates, etc. A tweet was labeled as an event if the information in it was sufficiently explicit and had a significant importance. Neutral events were considered to be all other tweets. Because of the nature of manual labeling, only the 1000 fastest growing threads were labeled and the final set was composed solely of 820 tweets whose labels the annotators agreed upon.

According to the authors, the developed FSD system was not evaluated because the authors already had an idea how the system performed through a previous testing with TDT5 data. Comparison with another state-of-the-art system (UMass (Allan et al. 2000)) revealed that the FSD system had equal event detection capability and a decreased processing time. Another reason for not performing a system evaluation was that evaluating a system of this scale would be extremely expensive because of the manual annotation.

Evaluation was done by computing the average precision on the 820 tweets and these were sorted according two criteria: (1) only event tweets are taken as relevant, and (2) event tweets and neutral tweets are taken as relevant. Table 2.4 shows the average precision for the different criteria using different types of ranking. The rankings used were: the baseline, which uses a random ordering of threads, the size of threads, the number of users and entropy + number of users (if the entropy is above a threshold then the thread is ordered by number of unique users).

The results showed that, for the relevant events, any of the used ranking methods performed better than the baseline, with the ranking that uses entropy outperforming all others. For events and neutral taken as relevant, the results were much better than the previous criteria and it can be seen that the baseline outperformed both the size of the thread ranking and the number of users ranking.

In the end it was verified that using local sensitive hashing adapted to first story detection while using variance reduction significantly improves processing performance and enabled the detection of

Ranking method	Events as relevant	Events/Neutral as relevant
Baseline	16.5	84.6
Size of thread	24.1	83.5
Number of users	24.5	83.9
Entropy + users	34.0	96.3

Table 2.4: Average Precisions

major events with precision and, with the use of entropy, it was able to reduce the amount of uninteresting information.

2.1.4 Automatic Event Classification Using Surface Text Features

(Hardy et al. 2006) present a data driven way of discovering events and their attributes that is part of a question answering system. This system is able to answer both factoid questions, whose answers can have multiple syntactic forms and can be represented in a finite list of answers, and analytical questions, that are broader and must be approached using different strategies.

(Hardy et al. 2006) use a Text Framing strategy to deal with analytical questions. The strategy imposes a partial structure to the text excerpt so that comparison, with other excerpts and with the question, is possible. General frames represent the topic of the text (e.g. pollution, trade, etc.), which is done by verifying the core verb, or noun, and named entities (e.g. person, date, etc.) of the excerpt. Typed frames represent specific events (e.g. transfer, assist, etc.) and their roles (e.g. destination, source, etc.).

Events were extracted from a corpus of 37444 documents from the Center for Non-Proliferation Studies and from a 178015 documents that were extracted from the web and covered topics related to weapons. A data driven approach was used in order to determine the most interesting events and their structure and then to extract them from the texts.

Using the BBN's IdentiFinder (Bikei et al. 1991), entities were tagged depending on the context they were integrated in, and considering the most common verbs and nouns in these contexts they were classified into different event types.

A total of 3996 event instances were identified and split into 10 event types (i.e. Agree, Assist, Attack, Develop, etc.) plus a None event type, which included all other events that did not fit the other event types. From these events, Transfer, Develop, Attack and Agree had 100 of their instances chosen to be analyzed and have patterns created for them, as they were the types with most instances. The patterns were designed according to syntactic elements of the text excerpts, such as keywords, prepositions, noun phrases and named entities. These handmade patterns are later compared with the

Algorithm	Number of Features			
	170	124	114	94
Logistic	59.13%	59.76%	59.46%	58.61%
Vote	58.98%	58.81%	58.56%	56.58%
Bagging	52.93%	52.93%	52.98%	52.33%
Baseline	21.45%	21.45%	21.45%	21.45%

Table 2.5: Classification results for the combined feature sets.

automatic classifiers.

Some textual elements that were considered to be important in the recognition of events (e.g. sequence length, part of speech and named entities) were encoded into features and then passed through an automatic classifiers.

Through the use of the Weka toolkit (Witten et al. 2011), three classifiers were selected, as well as a baseline that chose the majority class.

The first algorithm, called Logistic, builds a logistic regression model.The second, called Vote, is a meta algorithm that combines the results from four different base classifiers, including Naive Bayes, REP tree, Random Forest and Part. The third is called Bagging, that reduces variance of a classifier (REP tree).

From the 3996 annotated instances that comprise the 11 event types, 20 to 100 most frequent words were collected, excluding stop words, for the parts of speech that were considered to be most useful in the prediction of event types (i.e. noun, verb, adjective).

Applying event classification revealed that increasing the number of most frequent words also increased the accuracies of all algorithms, and that nouns provided the best accuracy percentages for every algorithm.

Tests were made using different combinations of features, totaling in 170, 124, 114 and 94 features, comprised of different amounts of nouns, verbs, adjectives, pronouns and named entities. Table 2.5, shows the accuracies of each algorithm, when faced with the different feature combinations.

It is possible to see that the Logistic algorithm yielded the best results with 124 features. Despite this, the overall accuracy of the classification algorithms decreases when a there is a reduction of the number of total features.

A closer analysis of the best results of the Logistic algorithm revealed that, of the 11 event types, Transfer, Develop, Attack and Agree had the highest number of detections (ignoring the None event, as it is uninteresting for this experiment).

Comparison of these results with a manual classification of the events, yielded the results seen in Ta-

Class	Number of Instances	Automatic Approach	Manual Approach
Transfer	857	0.689	0.4755
Develop	540	0.634	0.4518
Attack	687	0.711	0.5504
Agree	325	0.727	0.7134

Table 2.6: F-Scores of the two approaches

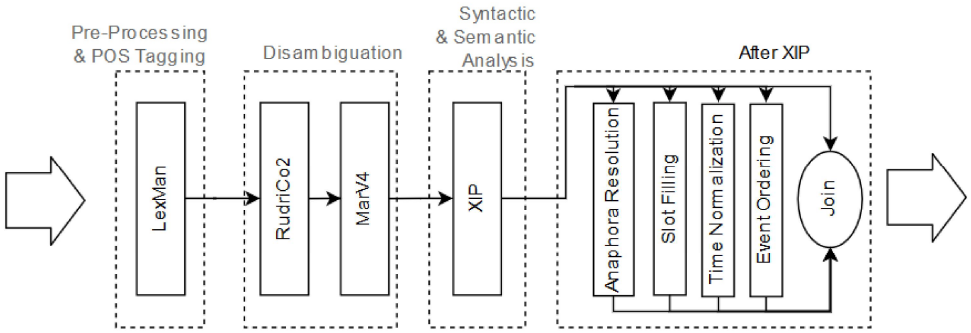

Figure 2.1: STRING architecture

ble 2.6. It was concluded that a machine learning approach can yield better results than the handcrafted patterns.

2.2 The STRING Architecture

The Statistical and Rule-Based Natural Language processing chain (STRING) (Mamede et al. 2012), developed at L2F, is composed by several different modules, shown in Figure 2.1, which are connected through the use of XML (Extensible Markup Language).

The first module of STRING is the LexMan (Lexical Morphological Analyzer) (Vicente 2013), which is responsible for the text segmentation and the morphosyntactic analysis.

The module divides the received input into sentences, which are then divided into tokens, and then assigns POS (Part-Of-Speech) tags to each token. The tag set is comprised of twelve categories (e.g. noun, verb, adjective, etc.) and eleven fields that encode information (e.g. category (CAT), subcategory (SCT), person (PER), gender (GEN), etc.).

During the segmentation phase, it identifies some notable tokens such as email addresses, numbers, abbreviations with a dot (cases where the dot does not mean the end of a sentence) and other symbols.

The assigning of tags is done with the use of finite-state transducers and, if it fails to assign a tag, it searches for similarities between the word in question and other known words and then assigns a proper tag.

In the end, the tagged segments are passed onto the next step of the STRING system, the disambiguation process, which is composed of two modules, the RuDriCo2 and the MARv4.

RuDriCo2 (Rule Driven Converter) (Diniz et al. 2010) is a rule-based morphological disambiguator. It receives the results of the previous step and attempts to resolve any existing conflict among the tags, changing them and the word segmentation accordingly. The segmentation modifications can be either contractions or expansions. For example, "Bem" ("Well"), "visto" ("seen"), that suffer contraction to form the single segment "Bem-visto" ("Well seen"). On the other hand, consider the segment "na", which undergoes an expansion in order to form the segments "em" ("in") and "a" ("the").

The second module performs statistical disambiguation and it is called MARv4 (Morphosyntatic Ambiguity Resolver) (Ribeiro 2003). The module analyzes the labels that were attributed to each token in the previous module, and chooses the best POS tag for each one using a Hidden Markov Model (Jurafsky and Martin 2017) and the Viterbi algorithm (Forney 1973). The language model used is based of trigrams (second-order models), which codify contextual information related to entities, and unigrams, which codify lexical information.

The last module of the STRING architecture is XIP (Xerox Incremental Parser) (Ait-Mokhtar et al. 2002) which is responsible for the syntactic analysis and dependency extraction from the results from the previous module. This module is further explained in Section 2.2.1.

The entire process ends with a post-syntactic analysis (After XIP) which is composed of four modules that deal with Anaphora resolution (Marques 2013), Slot filling (Carapinha 2013), Event ordering (Cabrita 2014) and Time expression normalization (Maurício 2011).

2.2.1 XIP System

XIP (Xerox Incremental Parser) (Ait-Mokhtar et al. 2002) is a rule-based parser developed by the Xerox Research Center Europe (XRCE). The Portuguese grammar, initially developed under a collaboration between L2F (INESC-ID Lisboa) and Xerox, has been enhanced in order to encompass many syntactic and semantic phenomena.

XIP is capable of applying some disambiguation rules to facilitate the parsing of information. These rules have the following structure:

```
filter = |left_context| selected_readings |right_context|
```

Where `filter` specifies a list of categories and features that are associated with a word and `selected_readings` corresponds to interpretations selected by the filter, which may have some optional context. Left and right context provide further precision to the application of the rules.

Data in XIP is represented as a node unit and has the following structure:

```
node:  category[features].
```

The `node` represents a specific word or expression and it contains a `category` (e.g. noun, verb, etc.) and a set of `[feature pairs]`. These features may represent various morphologic, syntactic or semantic information about the word.

XIP allows for the modification of its existing lexicon as well as the definition of new custom lexicons. Lexical rules provide a more precise interpretation of the tokens associated with a node and are represented in the following way:

```
lemma(:  POS([features])) (+)= (POS)[features].
```

The contents of the parenthesis of the rule change the functionality of the rule. `lemma = (POS)[feature]` will define a `lemma` as having a `POS` tag (e.g. noun, verb, etc.) with certain features.

Lexical rules may also be used in order to define additional information, of semantic or syntactic nature, which may be required in order to perform a correct parsing of the input. The addition of information requires the use of `+=` in the rule. For example, `lemma:  noun += [feature]`, where a lemma, which is a noun, gains a the feature in `[feature]`.

Through the use of chunking rules, sequences of categories are grouped into elementary syntactic structures, *chunks* (e.g. a NP is a chunk representing a noun phrase, which can be defined by grouping a noun and an article). The chunks have their heads identified, which will facilitate the analysis of dependencies. Chunking rules must be defined in a layered way, which defines the order in which they are executed. The chunking rules are divided into two types:

- the sequence nodes, that deal with ordered sequences of nodes;
- the immediate precedence and linear precedence rules (ID/LP), where the ID rules deal with unordered sets of nodes and the LP rules, with the help of the ID rules, establish some order between these nodes. An ID rule can be represented as such:

  ```
  layer> node -> list-of-lexical-nodes
  ```

Notice that the order of the `list-of-lexical-nodes` is irrelevant for the rule. For example, the rule:

```
1> NP -> det, noun, adj.
```

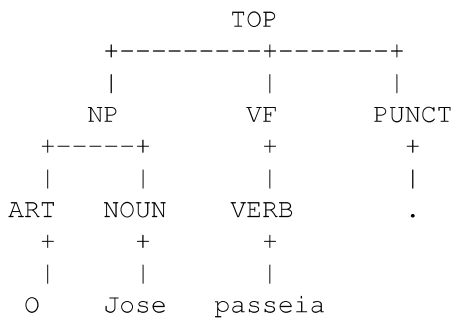

Figure 2.2: Chunking tree (simple example).

accepts any sequence of `det`, `noun` and `adj`. For the order of the word categories to be taken into consideration, it would be necessary to incorporate an LP rule, which would then allow the definition of optional categories, multiple instances or further constraining using left and right context.

The final result of the chunking process will be a (shallow) chunking tree that shows the basic relations between the word nodes. Figure 2.2 represents the chunking tree for the sentence "O José passeia." (José strolls.).

The next step in the parsing is the establishment of syntactic dependencies between chunks. The analysis of dependencies is done through the use of dependency rules, which represent the syntactic relations between chunks and between elements of the chunks. They can also be used to extract dependencies between the chunk heads, namely syntactic relations (e.g. subject, direct complement, indirect complement, etc.). Dependency rules can also be used to build other kinds of relations that lead to the identification and classification of temporal expressions or named entities, as well as, the identification of events.

The basic format of a dependency rule is as follows:

```
|pattern| if <condition> <dependency_terms>
```

A dependency rule is divided into three parts, the `|pattern|`, the `<condition>` and the `<dependency-terms>`. XIP uses Tree Regular Expressions (TRE) in order to build connections between distant nodes, which allows the pattern to be divided into segments that are not directly connected in the sentence that is being evaluated. This makes it so that a pattern may contain a TRE that describes the structural components of other parts of the input tree. The `condition` can be any boolean expression that is built from the dependency terms and these terms have the form `name[features](args)`, where the name denotes a dependency relation followed by its features and respective arguments.

A usual dependency rule will have a similar structure to the following example:

```
DETD(Jose, O)
SUBJ_PRE(passeia, Jose)
EVENT_LEX(passeia, outro)
EVENT_OTHER(passeia)
EVENT_AGENT(passeia, Jose)
0>TOP{NP{O Jose} VF{passeia} .}
```

Figure 2.3: Calculated Dependencies

```
if( (SUBJ(#3[lemma:ser],#2[human])) &
(PREDSUBJ(#3,#4[gentcountry]))
EVENT[lex=+](#4,##noun#6[surface:nascimento])
```

This example can identify "O João é português." ("João is portuguese."). A very rough interpretation of the example could be that there is an event of birth, i.e. "O João nasceu em Portugal." ("João was born in Portugal."), that can be extracted from the sentence. Notice that, in these directives, the concepts of nationality and place-of-birth were not distinguished. If in

```
(SUBJ(#3[lemma:ser]#2[human]))
```

the variable with a feature of human `#2[human]` (representing "João") is the subject of the action `#3[lemma:ser]` ("é" ("to be")) and if in

```
(PREDSUBJ(#3,#4[gentcountry]))
```

`#4[gentcountry]` (representing "português" ("portuguese")) is the target of the action `#3[lemma:ser]`. With this, in `EVENT[lex=+](#4,##noun#6[surface:nascimento])`, `#4` will be associated with a virtual node filled with a noun "nascimento" (birth) `##noun#6[surface:nascimento]` that represents the entity.

Running the dependency rules generates a list of calculated dependencies that show the various dependencies that could be extracted between the various components of the sentence that was analyzed. The output result for the previous example can be seen in Figure 2.3.

- `DETD(Jose, O)` associates the subject "Jose" with the determinant "O".
- `SUBJ_PRE(passeia, Jose)` associates the subject "Jose" with the action of "passeia" ("strolls"). `_PRE` indicates that the subject is located to the left of the verb, in the sentence.
- `EVENT_LEX(passeia, outro)` associates the word "passeia" ("strolls") with the dummy subtype "outro" ("other"), which is a subtype for events that have yet to be properly defined.
- `EVENT_OTHER(passeia)` associates the word "passeia" with the event type `OTHER`, which a nondescript designation of an event.

- EVENT_AGENT(passeia, Jose) indicates that the word "Jose" is the agent that is responsible for the event.

These dependencies operate after the syntactic analysis, therefore, errors in the previous phases may produce errors in the event dependencies.

Notice that a group of auxiliary dependencies were omitted (e.g. the HEAD relations between components.).

2.2.2 Types of events that are already identified

The STRING chain is currently able of identifying two types of well defined major event categories: the LIFETIME category and the BUSINESS category. These events associate a certain occurrence with a person or named entity.

The LIFETIME event category encompasses some relevant events that happen throughout the lifetime of a certain person. Basic life events such as BIRTH (nascimento), DEATH (morte) and AGE (idade) associate a person with that specific occurrence. The category also encompasses the family relations (i.e. PARENT, SIBLING, GRANDPARENT, etc.) between two people. It includes a RESIDENCE (residência) event that associates the person in question with any location where that person as resided in and an EDUCATION (educação) event that associates the person with a degree that he/she has attained throughout his/hers lifetime.

The BUSINESS event category encompasses some relevant events and professional relations between people and organizations. The WORK event associates a person with an organization or a profession. The FOUNDATION event associates a person with the foundation of an organization and the OWNERSHIP associates a person of having ownership over an organization. The BUSINESS category also encompasses a CLIENT and an AFFILIATION relation between a person and organization.

Table 2.7 shows examples for the previously described events.

There are other categories of events that STRING is able to identify, to some degree, these being, CRIME, LOCATION and PUBLIC events. These events are the target of this work and will be further explored in the following chapter.

2.2.3 Auxiliary Dependencies

The process of building new event dependencies requires the existence of of some auxiliary dependencies that are used to build the new dependencies.

Category	Event Type	Example
LIFETIME	BIRTH	O Jorge nasceu a 25 de Maio.
	DEATH	O senhor Luís faleceu hoje de manhã.
	AGE	O Pedro tem 19 anos.
	FAMILY	A Maria é sobrinha de Pedro Nunes.
	RESIDENCE	A Fátima viveu 3 anos em Coimbra.
	EDUCATION	O João tirou a licenciatura no IST.
BUSINESS	WORK	O Pedro é engenheiro informático há 2 anos.
	FOUNDATION	Bill Gates é o fundador da Microsoft.
	OWNERSHIP	O Pedro possui acções da Vodafone.
	CLIENT	A Maria é cliente habitual do Jumbo.
	AFFILIATION	O João é sócio do Benfica.

Table 2.7: Lifetime and Business Events

These auxiliary dependencies are generated during the syntactic analysis of a sentence and relate to a specific syntactic element of the sentence.

STRING is capable of extracting syntactic elements such as the subject of the sentence, its direct and indirect complements, the predicative of the subject and other predicate modifiers.

Consider the following example:

- O Pedro deu uma flor à Maria.
  Pedro gave a flower to Maria.

The analysis of this sentence generates the following syntactic dependencies:

- `SUBJ_PRE(deu,Pedro)`
- `CDIR_POST(deu,flor)`
- `CINDIR_POST(deu,Maria)`

The first dependency indicates that Pedro is the subject of the predicate, i.e. dar (to give), of this sentence, the second dependency indicates that flor (flower) is the direct complement and the last dependency indicates that Maria is the indirect complement.

The `_PRE` and `_POST` suffixes indicate that the elements is either at the left (`_PRE`) or at the right (`_POST`) of the predicate in the sentence, this helps convey the order of the syntactic elements of the sentence.

Information relative to the subordinate and main clauses of a sentence is also extracted by the system. Consider the following example:

- O Pedro disse que tinha visto o filme.
  Pedro said that he saw the movie.

The analysis of this sentence generates the following dependencies related to the main clause and the subordinated clause of the sentence:

- `INTRO(que,visto)`
- `CONNECTOR(disse,que)`
- `QBOUNDARY(que,que,filme)`

The first dependency connects the verb of the subordinate clause to the subordinate conjunction of the sentence, the second dependency connects the verb of the main clause to that same conjunction and the last dependency connects the first and last elements of the subordinate clause.

In terms of coordinated clauses, the system extracts the `COORD` dependency, which extracts the relations between the clauses and the conjunctions. For example:

- O Pedro fugiu e escondeu-se.
  Pedro fled and hid.

  will generate:

- `COORD(e,fugiu)`
- `COORD(e,escondeu)`

At last, the system is also capable of extracting the named entities present in a sentence, through the use of the `NE` dependency. For example:

- O Pedro trabalha na Microsoft.
  Pedro works at Microsoft.

  will generate:

- `NE_PEOPLE_INDIVIDUAL(Pedro)`
- `NE_INSTITUTION(Microsoft)`

The combination of these different dependencies allow for the construction of the new event dependencies that are used to identify new events.

Table 2.8 shows theses syntactic elements and their respective formal representations.

Element	Dependency representation
Subject	SUBJ(<predicate>,<subject>)
Predicative of Subject	PREDSUBJ(<predicate>,<subject predicative>)
Direct Complement	CDIR(<predicate>,<direct complement>)
Indirect Complement	CINDIR(<predicate>,<indirect complement>)
Modifier	MOD(<predicate>,<modifier>)
Named Entity	NE(<entity>)

Table 2.8: Auxiliary Dependencies

2.3 Synopses

This section began by presenting four different experiments. Each of them presented their own different ways of dealing with the identification and classification of events in real world scenarios.

The very first experiment showed a rule-based approach to accomplish event extraction from tweets, where sets of rules were used to classify the each tweet on one of two types of events.

The other three experiments utilized different automatic approaches to achieve their goals. The first of these attempted to find the best way to identify events on a word level by exploring different types of text representation techniques. The second experiment attempted to identify tweets on a continuous stream of tweets, where the solution was to develop a program capable of filtering through large amounts of tweets, in an efficient way, and group them according to a similarity score. The final experiment tested 3 different algorithms and several types of text features in order to find the best combination possible to better classify events. After the experiments, the architecture and functionality of the STRING system was displayed, followed by an indication of the different types of events that were implemented prior to this work, and a description of auxiliary dependencies that are used in the construction of futures event dependencies.

3 Event Identification

This chapter presents, for each of the event types, a description of the situations that are considered as events, in the context of STRING. The reasoning behind each decision, the 'why' a situation is considered to be an event, while another situation is not, is presented and discussed.

The chapter begins with the description of the `Crime` related events, followed by the description of the `Location` related events and, finally, the description of the `Public` related events.

Then, follows a description of the relations that were chosen to extract the important types of information that better describe the the event that they belong to. Each relation extracts a key piece of the event associated with them. Events have core relations that must be extracted every time, otherwise the situation is not considered as being an event of that type. Some relations can also be extracted for different types of event.

The next section begins with the description of the relations belonging to the `Crime` related event, followed by the relations from the `Location` related events and ends with the description of the relations of `Public` related events.

The selection of the relations for each event is based on Baptista, Mamede, Cabarrão, and Dias (2018), which was updated for the purpose of this work.

3.1 Event Recognition

This section describes the way STRING recognizes the new events and, for each event, it shows which situation are taken into account for them to be considered a part of that event.

3.1.1 Crime Category

3.1.1.1 Crime

A crime can be considered to be any violation that is legally condemned by the law of a country/community.

The actions that may be considered to be criminal may differ from country to country. Therefore, this work focuses solely on the list of crimes that are defined in the Portuguese Penal Code (Código Penal Português) (Cod). For example, gambling is considered to be a crime in Taiwan, not in Portugal.

The fastest way to detect if a sentence contains a crime is to check for the existence of nouns that identify the crimes themselves:

- O Pedro cometeu o homicídio de 2 pessoas.
  Pedro committed the murder of 2 people.

Another way to do it is through the verbs that are contained in the sentence. Some verbs, i.e. roubar (t o steal), assassinar (to murder), etc., can directly showcase an intent to practice a crime, for example:

- O Pedro conspirou.
  Pedro conspired.

But usually, verbs must be accompanied with further context in order to better determine the presence of a crime, for example:

- O Pedro foi atacado com uma faca.
  Pedro was attacked with a knife.

This example is considered to be a crime, as the use of the sharp object faca (knife) can put in danger the physical integrity to the person. The lack of this last piece of information would not allow the identification of this event as containing a crime. For example:

- O Pedro atacou o homem com insultos.
  Pedro attacked the man with insults.

is not considered to be a crime because the person suffered no real harm.

Some types of action are only considered to be crimes when there is a clear human victim to the action:

- O Pedro matou o Jorge.
  Pedro killed Jorge.
- O Pedro matou a galinha.
  Pedro killed the chicken.

The first example demonstrates a clear criminal event where a person was killed by another but, the second event is not considered to be one.

One last way to recognize a crime is through the adjectives that characterize the subject of a crime. For example:

- O Pedro é o homicída do João.
  Pedro is the murderer of João.

In STRING, not all sentences that present a violation of the law are considered as being a crime, i.e. even if a sentence presents a criminal action, the system will ignore it. To consider a sentence for a criminal event, it must contain a textual element or elements that denote a criminal action and to have either a culprit responsible for executing that criminal action or to have a victim whose action was acted upon.

The following example:

- Um homem roubou um colar.
  A man stole a necklace.

Is an acceptable case of a crime event because it contains both the crime that was perpetrated (roubar um colar (to steal a necklace)) and the perpetrator of the crime (Um homem (A man)).

The following example:

- Multiplos carros foram roubados na passada terça-feira.
  Several cars were robbed in the past Monday.

Is an example of an event that is not considered to be a crime event in this representation because it fails to state who perpetrated the crime or who was the person that suffered from the occurrence of the crime.

3.1.1.2 Trial and Prision

A trial is a session where a judge or a tribunal carefully examines facts related to an infraction of the law in order to determine the innocence or guilt of a person.

In STRING, a situation is only considered to be part of the `TRIAL` event category if it directly relates to a crime, i.e. the sentence must explicitly present the crime that a person is in trial for.

- Um homem vai a julgamento por branqueamento de capitais.
  A man will go on trial for money laundering.
- O Pedro foi julgado pelas sua acções.
  Pedro was judge for his actions.

The first example passes as a valid event case, while the second fails for not presenting the criminal action.

If the sentence reveals the entity that passes the judgment them it is required that this entity is either a court or a citizen with authority (judge). For example:

- A população condena o indivíduo pela ameaça de bomba.
  The population condemns the individual for the bomb threat.

This example cannot be considered for a `TRIAL` event, as the entity passing the judgment does not hold any judicial power to do so.

The `Prision` event type, on the other hand, can be divided into two subtypes, the `detenção` (arrest) and the `prisão` (prison).

The `detenção` (arrest) subtype deals with situations where a person has been arrested for having committed, or being suspected of having committed a crime. A situation can only be considered part of this subtype if the person being arrested is yet to have been sent to prison. For example:

- Cinco homens foram detidos pela GNR por suspeitas de corrupção.
  Five men were arrested by GNR under suspicion of corruption.

On the other hand, the `prisão` (prison) subtype deals with cases where the criminal has already been convicted and is serving a sentence for a crime. For example:

- A assassina está a cumprir uma pena de prisão perpétua pela morte da criança.
  The murderer is serving a life sentence for the death of the child.

These two cases deal with the lawful confinement of criminals, although under different conditions.

As part of the `crime` event category, the `prision` event type requires the presence of a crime in a sentence in order to be properly identified as such. Hence, any cases such as:

- Dois homens foram detidos pela GNR ontem à noite.
  Two men were arrested by GNR last night.

- A pena de prisão de Catarina irá durar 6 anos.
  Catarina's prision sentence will last 6 years.

are not considered as events in STRING for lacking a `crime`.

3.1.2 Location Category

3.1.2.1 Static

The `static` event type deals with events that are related with the permanence of entities at a certain location.

- A Estátua da Liberdade está em Nova Iorque.
  The Statue of Liberty is in New York.

The previous sentence contains a `static` location event in which the location of the monument Estátua da Liberdade (Statue of Liberty) is being defined as being in Nova Iorque (New York).

The "permanence" of an entity at a location can be recognized through single verbs, e.g. estar (to be), ficar (to stay), or the combination of verbs, e.g. fica situada (is located):

- A Torre Pisa fica situada em Itália.
  Tower of Pisa is situated in Italy.

Or the combination of verbs with pronouns, e.g. encontra-se (is in):

- O António encontra-se na Microsoft.
  António is at Microsoft.

For these events to be recognized it is required that the location or place to be properly defined in the sentence. These locations can take the form of any named place or a building/monument.

It is also possible for these events to have a specified duration, for example:

- O António esteve 3 horas na Microsoft.
  António spent 3 hours at Microsoft.

In the last example, for those three hours, the person António stayed in the place Microsoft.

3.1.2.2 Visit

The `visit` event, on the other hand, deals with situations where a person or an entity, metonymicaly (Baptista et al. 2011) (Oliveira 2010) being treated as a person, moves to or "visits" a certain place.

The simpler way to identify a `visit` event is through the presence of verbs that indicate movement, e.g. viajar (to travel), visitar (to visit). For example:

- Maria Fernandes viajou para o Japão.
  Maria Fernandes travelled to Japan.

Nouns that indicate that the entity is going, passing through or acting as a passerby, e.g. férias (vacation), viajem (travel), also indicate that a sentence is of `visit` type. For example:

- O Pedro está de férias na Escócia.
  Pedro in on vacation in Scotland.
- O Nuno está numa viajem de negócio para a Rússia.
  Nuno is on a business trip to Russia.

3.1.3 Public Category

3.1.3.1 Public

A situation can be considered to be part of the `public` event type if it relates a human entity to any sort of public activity that gathers multiple people. For example:

- As festas no Terreiro do Paço reuniram muitas pessoas.
  The festivities in Terreiro do Paço gathered several people.

Five types of `public` events are considered in STRING, the `culto` (cult) type, which identifies all sorts of public activities that are related to religious or cultist activities, e.g. casamento (marriage), batizado (baptism), etc., the `art-session` type, which identifies relevant art related activities, e.g. peça de teatro (play), musical, etc., `reunião` (meeting), which identifies any sort of activity where multiple people gather to discuss any topic, e.g. conferência (conference), reunião (meeting), etc., `desfile` (parade), which identifies any sort of parade like activity of several people, e.g. desfile de moda (fashion parade), procissão (procession), etc. and `festa` (party), which identifies all festive activities, e.g. festa (party), banquete (banquet), etc.

Other situations that may be, initially, considered as a Public event, but cannot be integrated into these categories, are not considered as part of the Public event.

These events can be identified through the combination of a verb and noun that identifies the events, such has:

- O Pedro foi à festa no Barreiro.
  Pedro went to the party in Barreiro.

It is also possible to identify some of these events through a verb that describes the event, for example:

- Os senhores reuniram para discutir a situação.
  The gentleman have gathered to discuss the situation.

Sentences that do not explicitly indicate an entity that is participating in the public event are considered invalid and are not counted as part of the `public` event type. For example:

- O Pedro foi ver a festa.
  Pedro went to see the party.

3.1.3.2 Ephemeride

The `ephemeride` event is a type of public event that celebrates the anniversary of relevant events. For example:

- Celebra-se os 100 anos da abertura do IST.
  One celebrates the 100th anniversary of the opening of IST.

This event is divided into three subtypes. The two first event subtypes correspond to major life milestones in a person's life: `nascimento` (birth) and `morte` (death). For example:

- Fazem 100 anos do nascimento de Nelson Mandela.
  One celebrates Nelson Mandela's birthday.

One requirement of the `nascimento` (birth) and `morte` (death) subtypes is that the name of the person, whose life milestone is being celebrated, must be explicit in the sentence. Therefore the following example is not considered as a `ephemeride` event:

- Faz uma semana da morte de duas raparigas.
  It's been a week since the death of two girls.

The last event subtype is the `efeméride` which deals with the anniversary of any other relevant celebrations, such as inaugurations, treaties, historical events, etc. For example:

- Festejam-se os 29º aniversário da Queda do Muro de Berlim.
  The 29th anniversary of the Fall of the Wall of Berlin is celebrated.

Similarly to the previous two `ephemeride` subtypes, this one requires that the name of the entity is explicit in the sentence.

This is the only event in this work that does not require the presence of an individual to be identified.

The identification of this event is done through the identification of verbs that indicate the a celebration or festivity in the name of another event, e.g. celebrar (to celebrate), festejar (to party), followed by the event in question.

3.2 Event Relation Formulation

The key goal of this work is to obtain information relative to the events through the analysis of each event occurrence.

Each event can extract different kinds of information exclusive to that event that reveals important and useful information.

Some kind of information are not exclusive to one kind of event. Among every event it is possible to extract information relative to the time when the event took place and the location where it happened. This information is identified by the event relations of `DATE` and `PLACE` and exist for every kind of event explained in this document. In the `Location` and `Public` event categories, the `DATE` relation is also divided into other relations that reveal more specific temporal information, such as the beginning, end and duration of the event. For example:

- O Pedro ficou 3 horas em Lisboa.
  Pedro stayed 3 hours in Lisbon.

```
EVENT_DURATION(ficou,3 horas)

EVENT_PLACE(ficou,Lisboa)
```

- O Pedro esteve na reunião que acabou às 15:30.

  Pedro was at the meeting that endend at 3:30 PM.

  ```
  EVENT_DATE-END(reunião,15:30)
  ```

The following sections describe the relations that are specific for each of the event categories. Note that the following sections assume that the EVENT_LEX and the EVENT_<TYPE> (i.e. CRIME, TRIAL, PRISION, STATIC, VISIT, PUBLIC and EPHEMERID) dependencies have already been extracted in a regular analysis and are considered to require no further description, for they are core in the extraction of new events.

3.2.1 Crime Specific Relations

Relation	Description
AGENT	The perpetrator of the crime
VICTIM	The victim of the crime
INSTRUMENT	The instrument used to practice the crime
OBJECT	The object taken without consent (stolen/smuggled)
DATE	The time when the crime takes place
PLACE	The location where the crime takes place

Table 3.1: Relations for the Crime event type

Table 3.1 shows the relations that can be extracted for the crime event. For a sentence to be considered as having a crime event it requires the presence of a culprit, a suspect or a victim. For example:

- O Pedro roubou o carro.

  Pedro stole the car.

- O carro da Maria foi roubado.

  Maria's car was stolen.

In the first example, the relation AGENT is extracted in order to capture the person that is responsible, or is suspected of being responsible, for the crime, which, for the previous example, is represented in the following way:

```
EVENT_AGENT(roubou,O Pedro)
```

The VICTIM relation, on the other hand, captures the human entity that suffered directly from the crime, generating, for the second example, the following relation:

```
EVENT_VICTIM(roubado,Maria)
```

From each of the previous examples, it is also possible to extract another relevant information, which is the item that was stolen and is identified through the OBJECT relation. This relation can only be extracted if in situation where an object is taken from its rightful owner, against his will.

Another relation that can be extracted from a crime event relates to the use of a weapon to carry out the crime. This relevant information is extracted through the INSTRUMENT relation.

Consider the following example and the respective extracted relations:

- No dia anterior, um homem usou uma faca para roubar as carteiras de um grupo de 3 adolescentes.

  In the previous day, a man used a knife to rob the wallets of a group of 3 adolescents.

```
EVENT_AGENT(roubar,um homem)

EVENT_VICTIM(roubar,Um grupo de 3 adolescentes)

EVENT_INSTRUMENT(roubar,uma faca)

EVENT_OBJECT(roubar,carteiras)

EVENT_DATE(roubar,Em o dia anterior)
```

This example shows the proper extraction of the relevant relations and the order of how they must be extracted. First, the AGENT and VICTIM relations, which are the obligatory relations for this event, then the other event specific relations and, finally, the generic relations, i.e. DATE or PLACE.

3.2.2 Trial Specific Relations

Relation	Description
DEFENDANT	The person under trial
COURT	The court where the trial is held
JURISDICTION	The jurisdiction of the court
OUTCOME	The outcome of the trial
SENTENCE	The sentence to be served
FINE	The fine to be payed
DATE	The time when the trial takes place
PLACE	The location where the trial takes place

Table 3.2: Relations for the Trial event type

The extraction of information relative to the a trial event requires the previous extraction of the crime event that describes the law infraction that is the cause for the trial event. Table 3.2 shows the relations that can be extracted for this event.

The trial event extracts extracts information relative to the conditions of the trial and has only one obligatory relation, the DEFENDANT, which identifies the entity that is under trial for the crime that was identified in the crime event.

Consider the following example:

- Três homens estão em julgamento por homicídio.
  Three men are on trial for murder.

The analysis of this sentence generates the crime related relations, of which the AGENT relation identifies the culprit that committed the crime. Then the extraction of the trial relations generates the DEFENDANT relation that indicates that the same culprit is the entity being trialed. The DEFENDANT usually coincides with the AGENT from the crime event type.

Other information that is considered to be relevant deals with the identification of the entity that evaluating the case, namely the court, which is extracted through the COURT relation. This relation allows the extraction of the additional relation JURISDICTION, which indicates the operational radius of the court in question. For example:

- O Tribunal Judicial do Porto condenou Norberto Martins pelo crime de homicídio qualificado.
  The Porto Judicial Court convicted Norberto Martins for the crime of qualified homicide.

```
EVENT_COURT(condenou,Tribunal Judicial)

EVENT_JURISDICTION(condenou,Porto)
```

STRING extracts the COURT relation, identifying Tribunal Judicial (Judicial Court) as the court responsible for evaluating the crime and the JURISDICTION relation extracts identifies Porto as being the jurisdiction.

Information relative to the outcome of the trial is extracted through the OUTCOME relation. The outcome of the trial can be either condenação (conviction), which indicates that the DEFENDANT was declared guilty, or absolvição (absolution), which indicates that he was declared innocent. For example:

- António Marques foi declarado inocente de todas a acusações de branquamento de capitais.
  António Marques was declared innocent from all accusations of money laundering.

```
EVENT_OUTCOME(declarado,absolvição)
```

- Os assassinos foram condenados a quinze anos de cadeia.
  The murderers were convicted to fifteen years in prison.

```
EVENT_OUTCOME(condenados,condenação)

EVENT_SENTENCE(condenados,quinze anos de cadeia)
```

If the outcome of the event is `condenação` (conviction), it is possible to extract the sentence that was applied to the `DEFENDANT`, after the `OUTCOME` was identified as being `condenação` (conviction). This information is extracted through the `SENTENCE` relation and can only be fulfilled when the `OUTCOME` is set to `condenação` (conviction).

Additionally, and depending on the sentence, it is possible for the `FINE` relation to be extracted. This relation indicates the monetary value that the `DEFENDANT` must pay as compensation for having committed the crime. For example:

- James Rodriguez foi condenado a pagar 11 milhões de euros por fraude fiscal.

  James Rodriguez was ordered to pay 11 million euros for tax fraud.

```
EVENT_OUTCOME(condenado,condenação)

EVENT_SENTENCE(condenados,pagar)

EVENT_FINE(condenado,11 milhões de euros)
```

3.2.3 Prision Specific Relations

Relation	Description
AGENT	The entity that made the arrest
DETAINEE	The person being arrested
PRISONER	The person being imprisoned or serving a sentence
SENTENCE	The sentence to be served
DATE	The time when the arrest/imprisonment takes place
PLACE	The location where the arrest/imprisonment takes place

Table 3.3: Relations for the Prision event type

As part of the `Crime` event category, the `prision` event type, similarly to the `trial` event type, requires the previous extraction of the `crime` event that is the reason of arrest or imprisonment. The relations that can be extracted for this event are shown in table 3.3

The `prision` event can be divided into two distinct subtypes, the `detenção` (detention) subtype, that deals with situations where an entity is under arrest but is yet to serve a sentence or penalty for a crime, and the `prisão` (prison) subtype, that deals with situations where the entity serves that sentence or penalty.

Considering these definitions, the DETAINEE relation identifies the entity that is being arrested for the previous extracted crime, and the PRISONER relation identifies the entity that is serving a sentence or penalty for that crime. These are the obligatory relations for the PRISION event and usually coincide with the AGENT relation from the crime event type. For example:

- A GNR deteve um homem foi detido por homicídio.
  GNR has arrested a man for homicide.

  ```
  EVENT_DETAINEE(detido,um homem)
  ```

- Pedro cumpre pena de 3 anos de prisão por tentativas de homicidio.
  Pedro is serving prison terms of 3 years for attempted homicide.

  ```
  EVENT_PRISONER(prisão,Pedro)
  ```

For the first example, a relation AGENT, of the crime event, identifies um homem (A man) as the perpetrator of the crime which is the same entity identified by the DETAINEE relation. For example:

```
EVENT_AGENT(homicídio, um homem)

EVENT_DETAINEE(detido, um homem)
```

Similarly, for the second example, the relations of AGENT, from the crime event, and the PRISONER relation from the prision, identify Pedro.

```
EVENT_AGENT(tentativa de homicidio, Pedro)

EVENT_DETAINEE(detido, Pedro)
```

It is also possible to extract information relative to the entity that is making the arrest, as is the case for the first example. The AGENT relation of the prision event, not to be confused with the relation of the crime event with the same name, identifies this entity, which, for the example, is A GNR. For example:

```
EVENT_AGENT(detido,GNR)
```

In addition, for the second example, information relative to the sentence that the PRISONER is serving, is extracted through the SENTENCE relation:

```
EVENT_SENTENCE(prisão,3 anos)
```

3.2.4 Static Location Specific Relations

The identification of a static location event is tied to the recognition of two key pieces of information: the entity whose location is being defined and that same location.

The static event can identify the locations of both objects, i.e. human constructs or natural sites, and human entities. The processing of these two kinds of entities is done through two different relations, which can be seen in table 3.4.

Relation	Description
PARTICIPANT	The human entity whose location is being defined
OBJ	The object whose location is being defined
MET-PLACE	The location where the event takes place (human entity being treated as a place through metonymy)
PLACE	The location where the event takes place
DATE	The time when the event takes place
DATE-START	The time when the event starts
DATE-END	The time when the event ends
DURATION	The duration of the event

Table 3.4: Relations for the Static location event type

If the entity is considered an object, the OBJ relation is used to identify that entity, for example:

- A Torre Pisa está situada em Itália.

  The Leaning Tower of Pisa is located in Italy.

  ```
  EVENT_OBJ(situada,A Torre Pisa)
  ```

Otherwise, the PARTICIPANT relation is used to extract a human entity, for example:

- O Pedro está na Gulbenkian.

  Pedro is at Gulbenkian.

  ```
  EVENT_PARTICIPANT(está,Pedro)
  ```

The PARTICIPANT relation is also used if the entity possesses attributes that allow it to be treated as a human through metonymy (Baptista et al. 2011)(Oliveira 2010), as is the case of the following example:

- A Microsoft está, esta semana, em Lisboa.

  This week, Microsoft is in Lisbon.

  ```
  EVENT_PARTICIPANT(está,Microsoft)
  ```

It is also possible to make distinctions relative to the location where the PARTICIPANT/OBJ is. Apart from the usual PLACE relation that extract the location of the entity, there is a MET-PLACE relation that is used in situations where the actual location is in fact an *organization* or *institution* that is treated as a place, metonymicaly (Baptista et al. 2011)(Oliveira 2010). For example:

- O José está agora no Continente.

  José is at Continente right now.

  ```
  EVENT_MET-PLACE(está,Continente)
  ```

For a sentence to be considered as a `static` location event, it is necessary that, at least, one of the `PARTICIPANT` or `OBJ` relations, as well as, one of the `PLACE` or `MET-PLACE` be present in the sentence.

3.2.5 Visit Location and Public Specific Relations

Relation	Description
`PARTICIPANT`	The participant of the event
`MET-PLACE`	The location where the event takes place (human entity being treated as a place through metonymy)
`PLACE`	The location where the event takes place
`DATE`	The time when the event takes place
`DATE-START`	The time when the event starts
`DATE-END`	The time when the event ends
`DURATION`	The duration of the event

Table 3.5: Relations for the Public and Visit location event type

Both the `visit` location event and the `public` event share the same kinds of relations, which can be seen in table 3.5. The differences are that the `visit` requires the information relative to the entity whose location is being defined, as well as that location itself, while the `public` event only requires the presence of, at least one participant.

The relation that extracts information relative to the entity that is the participant of each of these event is the `PARTICIPANT` relation, and can only extract information for human or human-like entities, i.e. non-human entities that are metonymically treated as humans for the sake of classification. For example:

- A Maria visitou a Nova Zelândia.

  Maria visited New Zealand.

  ```
  EVENT_PARTICIPANT(visitou,Maria)
  ```

This relation must be present in both types of events.

Information relative to the location of the event is extracted through a `PLACE` relation or through a `MET-PLACE` relation, this last one, similarly to the `static` event relation of the same name, is used when the place is, in fact, an *organization* or an `institution`.

- A Maria visitou o Instituto Superior Técnico.

  Maria visited Instituto Superior Técnico.

  ```
  EVENT_MET-PLACE(visitou,Instituto Superior Técnico)
  ```

At least either the PLACE or MET-PLACE relation must be extracted, in the visit location event, for the event to be valid.

3.2.6 Ephemerids Specific Relations

Relation	Description
ARG	The argument of the celebrated event
ELAPSED-TIME	The time elapsed since the event first happened
PARTICIPANT	The person participating in the event
PLACE	The location where the event takes place
DATE	The time when the event takes place
DATE-START	The time when the event starts
DATE-END	The time when the event ends
DURATION	The duration of the event

Table 3.6: Relations for the Ephemerid event type

As a special case of the Public event category, the ephemerid event type contains several new relations that are not present in the public event. The relations for this event are shown in table 3.6.

The ephemerid is the only event type that does not require the presence of a participating entity to be extracted but, when present, this information can be extracted through the PARTICIPANT relation. For example:

- O Pedro esteve nas comemorações dos 100 anos do IST.

  Pedro was at the celebration of the 100 years of IST.

  ```
  EVENT_PARTICIPANT(esteve,Pedro)
  ```

The ARG relation extracts the entity being celebrated. If the ephemerid event is of type nascimento (birth) or of type morte (death), ARG extracts the name of the person being celebrated. In some certain cases of the efeméride subtype, this relation can also extract information related to the celebrated entity. For example:

The extraction of the ARG relation is obligatory for the nascimento (birth) and morte (death) subtypes, and for some cases in the efeméride subtype where the celebration is not related to dates. For example:

- Hoje marca a data da morte de Michael Jackson.

  Today marks the date of the death of Michael Jackson.

  ```
  EVENT_ARG(morte,Michael Jackson)
  ```

- Celebram-se os 100 anos da abertura do IST.

  The 100 years of the opening of IST are celebrated.

  ```
  EVENT_ARG(abertura,IST)
  ```

In the first example, the `ephemerid` event is of type `morte` (death), which extracts the `ARG` relation identifying Michael Jackson as the person that died, while, in the second example, the `ARG` relation identifies IST as the target for celebration. The third example shows a situation where the `ARG` relation is not required, as Dia Mundial da Criança (World Children's Day) is the event itself and does not need further identification.

In the second example, it is still possible to extract one more piece of information, that is the time that has passed since the event first occurred. This information is extracted through the `ELAPSED-TIME` relation. For example:

```
EVENT_ELAPSED-TIME(abertura,100 anos)
```

4 Development

This chapter describes the methods used to expand the functionality of STRING. It begins with a description of the reasoning behind the new organization of the file system, then follow with a description of the migration of the old code to the new specialized files. Then follows a segment related to the development of dependency rules and the description of new improvements for their structure. Finally, the chapter ends with a description of the changes done to the lexicon pool, as well as an accounting of the total number of rules created during the development of this project.

4.1 File Organization and Migration

The STRING file system is composed of several dependency files that store the rules that are used during the processing of new inputs. Therefore, there are files dedicated to storing the rules that identify and classify events.

In order to store the rules that identify the new event categories, two new dependency files where created, i.e. for `location` and `public` events (the `crime` category already had a dedicated file). Dependency files have the following nomenclature:

```
dependencyEvent<TYPE>.xip
```

where `<TYPE>` denotes the event stored in that dependency.

Then it was necessary to migrate the already existent rule set to their respective files. The rules that were directed to the identification of `location` and `public` events were present in the dependency file that should be dedicated to the `lifetime` event category.

The migration phase was then initiated, and the rules were migrated accordingly to the `dependencyEventLocative.xip`, for the `location` category of events, and to the `dependencyEventPublic.xip`, for the `public` category of events.

During the migration it was necessary to pay special attention to rules that were shared among the events that were being migrated and the events in the `lifetime` dependency file. Rules that extracted relation of `DATE` and `PLACE` are some of the rules that are shared among all event dependencies but, for

the sake of preventing generation of relations out of order and wrongful generations, it was necessary to isolate these rules for each of the new event types.

This process was unnecessary for the case of the `crime` event category, as it already possessed its own dedicated dependency file and its rules were already separated from the other events.

4.2 Rule Development

This section will describe the methods used during the construction of new rules, as well as, describe the processes used in order to facilitate and improve these new rules, which lead to an easier understanding and maintenance of the rule sets.

4.2.1 Rule Construction

The process of creating new rules is dependent on the proper generation of the required syntactic rules, described in Section 2.2.3. The combination of these rules is the key to generate new dependencies.

Consider the following example:

- O homem roubou o dinheiro.
  The man stole the money.

```
SUBJ_PRE(roubou,homem)

CDIR_POST(roubou,dinheiro)
```

The analysis of the previous sentence generates a `SUBJ` dependency, identifies the subject of the sentence as being homem (man) and the `CDIR` dependency identifies the dinheiro (money) as being the direct complement of the sentence.

These dependencies are then used to construct rules that extract the relations that are related to the event in cause. For example, the two previous syntactic dependencies can are enough to extract a `crime` event, which generates the following output:

```
EVENT_LEX(roubou,roubo)

EVENT_CRIME(roubo)

EVENT_AGENT(roubou,o homem)

EVENT_OBJECT(roubou,o dinheiro)
```

The first event dependency that must be generated is LEX, as it associates the word that is the core of the event in question, with the event subtype.

The rule that generates this dependency has the following structure:

```
if(SUBJ[PRE](#1[lemma:'roubar'],#2[human]) &

CDIR[POST](#1,#3))

EVENT[LEX=+](#1,##noun#6[surface:roubo])
```

For this case, the if condition verifies the existence of the SUBJ (subject) and CDIR (direct complement) expressions where, for the SUBJ expression, the first argument #1 contains the feature [lemma:'roubar'] and that the second argument #2 contains the feature [human], and the CDIR expression contains the same first argument and a third argument (#3) with no relevant features. After these conditions are met, the rule may generate the required LEX dependency.

This process is similar for every subsequent rule but, it is possible for these rules to use the previously generated event related dependency to aid the extraction of their relations. For example:

```
if(EVENT[LEX](#1,#6[surface:roubo]))

EVENT[CRIME=+](#1)
```

This last rule only uses the previously generated EVENT_LEX dependency to generate the appropriate <TYPE> relation.

4.2.2 Rule Order

Inside each of the new event dependency files, the rules are written in an orderly way: first comes the EVENT_LEX rules which associate the trigger word with the event subtype, then comes the EVENT_<TYPE> which associates that trigger word with the event type and then come the event specific relations.

Consider the following sentence:

- Dois homens roubaram medicamentos de uma farmácia, na passada terça-feira.
  Two men stole medicine from a pharmacy in the past Tuesday.

The analysis of this sentence generates the EVENT_LEX and EVENT_<TYPE> rules in the following way:

```
EVENT_LEX(roubaram,roubar)

EVENT_CRIME(roubaram)
```

This order is key for every type of event and no event can be extracted if these first two rules fail.

The order between the rules that identify the relations of the event are, usually not relevant, and can be executed as soon as `EVENT_LEX` and `EVENT_<TYPE>` are extracted.

Some exceptions to this, are rules for relations that are dependant of other relations. Two examples of this are the `OUTCOME` and `SENTENCE` relations and the `COURT` and `JURISDICTION` relations of the `Trial` event. In the first case, the rules relative to the `OUTCOME` must be generated prior to the generation of the rules for `SENTENCE` and, in the same way, the rules for `COURT` must be generated prior to `JURISDICTION`. For example:

- O arguido foi condenado a 16 anos de prisão por homicídio qualificado.

  The defendant was sentenced to 16 years in prison for qualified homicide.

```
EVENT_LEX(condenado,julgamento)

EVENT_TRIAL(condenado)

EVENT_DEFENDANT(condenado,O arguido)

EVENT_OUTCOME(condendado,condenação)

EVENT_SENTENCE(condenado,16 anos de prisão)
```

Another important note on the order of execution of the rules is that, for rules from the `Crime` category, i.e. `crime`, `trial` and `prision`, it is necessary for the `trial` and `prision` to be executed **after** all the rules related to `crime` type. For example:

- Joaquim foi detido ontem, pela GNR, pelo homicídio que ocorreu na sexta-feira.

  Joaquim was arrested yesterday by the GNR for the murder that occurred Friday.

```
EVENT_LEX(homicídio,homicídio)

EVENT_CRIME(homicídio)

EVENT_AGENT(homicídio,Joaquim)

EVENT_DATE(homicídio,sexta-feira)

EVENT_LEX(detido,detenção)

EVENT_PRISION(detido)

EVENT_AGENT(detido,GNR)

EVENT_DETAINEE(detido,Joaquim)

EVENT_DATE(detido,ontem)
```

The previous example shows a well structured output where the information related to the `prision` event comes after the information for the `crime` event.

4.2.3 Rule Structure Improvement

During the rule construction phase, different ways of building dependencies were explored. A regular dependency is built in the following way:

```
if((SUBJ[PRE](#1[lemma:'matar'],#2[human]) ||
SUBJ[PRE](#1[lemma:'assassinar'],#2[human])) &
CDIR[POST](#1,#3[human]))
EVENT[LEX=+](#1,##noun#6[surface:'homicídio'])
```

This rule is able to identify both the sentence O Pedro matou a pessoa. (Pedro killed the person.) and O Pedro assassinou a Pessoa. (Pedro murdered the person.).

Attempting to create a rule that is able to identify all the different ways the same event can be conveyed is a strenuous task that may lead to a very large rule set.

The first attempt into simplifying the rules, was to verify the features of the duplicate arguments, i.e. `#2[human]` in the previous example, outside the logic 'OR' (`||`) comparisons. This generated the following structure:

```
if((SUBJ[PRE](#1[lemma:'matar'],#2) ||
SUBJ[PRE](#1[lemma:'assassinar'],#2)) &
#2[human] &
CDIR[POST](#1,#3[human]))
EVENT[LEX=+](#1,##noun#6[surface:'homicídio'])
```

This facilitated the writing of the rule by reducing the number of times an argument was defined for each condition.

The next step in the simplification was to find a way to evaluate the syntactic expressions a minimal number of times possible. The ideal was to reduce this evaluation to one per expression. For example:

```
if(SUBJ[PRE](#1,#2) &
CDIR[POST](#1,#3))
```

But, to do that, it is necessary to push the non-duplicate arguments to outside the expressions, as such:

```
if(SUBJ[PRE](#1,#2) &

CDIR[POST](#1,#3) &

(#1[lemma:'matar'] ||

#1[lemma:'assassinar']) &

#2[human] &

#3[human])
```

Notice the two arguments with feature `[human]`, each refer to different arguments with the same trait of human.

In case of single feature arguments, e.g. `#2` and `#3`, they can remain in the expressions, in order to reduce clutter, resulting in the following final rule:

```
if(SUBJ[PRE](#1,#2[human]) &

CDIR[POST](#1,#3[human]) &

(#1[lemma:'matar'] || #1[lemma:'assassinar']))

EVENT[LEX=+](#1,##noun#6[surface:'homicídio'])
```

This way of building new rules results in a far less cluttered rule set which, in turn, facilitates expansion, interpretation and error detection. This was implemented in the files related to the `Location` related events and `Public` events.

Another method of reducing the size of subsequent rules is to tie them to the core dependencies `EVENT_LEX` or `EVENT_<TYPE>`. For example:

```
if(EVENT[CRIME](#1[lemma:'roubar']) &

SUBJ[PRE](#1,#2[human]))

EVENT[AGENT=+](#1,#2)
```

This ties the rules to the specific event `<TYPE>`, by not allowing for the generation of the dependency if the core dependency have not been generated, and it reduces the dependencies required to assess the validity of the rule.

This method also functions as a guarantee that shared rules, such as, the rules for the `PLACE` and `DATE` relations are only executed inside the right event.

In total, the dependency construction effort resulted in the creation of 608 rules, spread over 329 rules for the `crime` event category, 99 rules for the `location` event category and 180 for the `public` event category.

4.3 Lexicon Update

The last major effort in improving the identification of new events was the development of the existing lexicon collection.

The lexicons are grouped into several different files, each dedicated to a concept, e.g. `human`, `nationality`, `brand`, etc.

The lexicon collection made a large contribution on the identification of crimes, aggressive actions, objects (both used in `crime` for the `INSTRUMENT` and `OBJECT` relations), buildings and locations (for the `location` events and `PLACE` relation), time, dates, public events, important dates and people.

The system possessed a lexicon collection to aid the development of events for the `crime` category, which leads to a very small increment of 38 new criminal action to the substantial pool of 372 criminal actions.

No new additions to the lexicon were necessary to improve the recognition of events of the `location` category, as there already exists a rather large pool for location such as, countries, cities, cultural sites, etc.

For the `public` event category, a few cases were added that aided the identification of each of the `public` event subtypes and a addition of 438 new entries to important dates was also added, which helped in the identification of the `ephemerid` event.

5 Evaluation

This chapter begins with the presentation of the corpus used for the evaluation of the systems performance, followed by the description of the setup and the methods used during the process of evaluation.

The metrics of precision, recall and f-measure are then used to measure the performance of the implementation of each of the events developed.

The results of the evaluation are then presented and discussed.

5.1 Corpora

In order to evaluate the performance of the implementation, a non-annotated corpus, extracted from the recordings of the Portuguese Parliament, was used.

It is organized into thirteen major directories, each with their own chain of directories. The directory chain is composed of 4030 directories that contain a combined total of 1227586 files which, in turn, contain a combined total of 115605895 words.

Each of the major directories represent different month where there was, at least, one Parliament session. These directories, each contain their own directory chain whose directories represent a different session or topic that was discussed.

The ending directories of the chain represent a particular topic and each file in it represents a continuous dialogue of an individual. Interruptions in the dialogue of a person were registered through the creation of a new file, i.e. every time someone was interrupted, a new file with the dialogue of that new person was created.

5.2 Evaluation Setup

In order to evaluate the newly developed events, a set of sentences was extracted from the given corpora.

Sentences were extracted from the corpora through the use of Python scripts that uses regular expression and key words in order to extract appropriate sentences for each of the event types.

For each of the seven event types, the script extracted all the entries that were found through the regular expressions, then, through the use of bash commands, 50 sentences were randomly selected for evaluation, culminating in a total of 350 sentences. The number of events in each group of 50 sentences, for each of the seven event types, is displayed in Table 5.1.

	Crime	Trial	Prision	Static	Visit	Public	Ephemerid
Events	25	23	19	8	18	11	37

Table 5.1: Number of events in the extracted sentences.

The extracted sentences were then ran through the original STRING system in order to provide a baseline on the number of correctly detected events for the evaluation. This baseline can be seen in Table 5.2.

	Crime	Trial	Prision	Static	Visit	Public	Ephemerid
Detected Events	13	4	0	0	0	0	3

Table 5.2: Baseline for the evaluation.

Then, the same sentences were ran through the modified STRING, which generated a file containing the full output of those sentences. These outputs were then stripped of the dependencies that were not related to the events present in the sentence (in case there was no event, all dependencies were removed).

For each of the outputs that were extracted, a corrected output file, with the correct output, was made manually. This file with the corrected output was compared with the original output in order to evaluate the performance of the system.

5.3 Evaluation

5.3.1 Methods for Evaluation

During the evaluation of the system, two aspects of event identification are taken into consideration:

- the identification of the event type
- the correct identification of the relations of the correctly identified events

Each step of the evaluation process evaluates a key part of the extraction of events from sentences.

The identification of the event types may be verified through the evaluation of the EVENT_LEX and EVENT_<TYPE> dependencies which identify the type of the event. Failure to extract any of these two dependencies equates to a failure in the identification of the event type, for example:

- if the EVENT_LEX dependency fails then the EVENT_<TYPE> is not extracted:

  EVENT_LEX(rapto,outro)

- the EVENT_<TYPE> may be extracted for the wrong event:

  EVENT_LEX(acusado,julgamento)

  EVENT_CRIME(acusado)

The extraction of relations helps verify the degree of which the system is able to identify important information for each event. Different types of errors may appear during the extraction of the relations:

- missing relations:
  - may generate:

    EVENT_LEX(preso,prisão)

    EVENT_PRISION(preso)

    EVENT_PRISONER(preso,o homem
  - instead of:

    EVENT_LEX(preso,prisão)

    EVENT_PRISION(preso)

    EVENT_PRISONER(preso,o homem)

    EVENT_DATE(preso,dia 3 de Setembro)

- too many relations:
  - may generate:

    EVENT_LEX(preso,detenção)

    EVENT_PRISION(preso)

    EVENT_DETAINEE(preso,o homem

    EVENT_AGENT(preso,o homem)
  - instead of:

    EVENT_LEX(preso,detenção)

    EVENT_PRISION(preso)

    EVENT_DETAINEE(preso,o homem)

- extraction of the wrong relation:
  - may generate:

```
EVENT_LEX(preso,detenção)

EVENT_PRISION(preso)

EVENT_DETAINEE(preso,o homem
```

– instead of:

```
EVENT_LEX(preso,detenção)

EVENT_PRISION(preso)

EVENT_AGENT(preso,o homem)
```

5.3.2 Metrics

During the evaluation the results are stored in confusion matrices that gives an overview of the performance of the system. Table 5.3 show the organization of a confusion matrix for evaluation of events. Similar matrices are also used for the evaluation of relations.

Predicted\Actual	Event	No Event	
Event	True Event	False Event	Events Predicted
No Event	False No Event	True No Event	No Events Predicted
	Actual Events	Actual No Events	Total

Table 5.3: Confusion Matrix for events

The columns of the matrix show the number of existing sentences with an event and without an event, while the rows show the number of sentences that were identified as having an event and not having an event.

The results that are obtained through the confusion matrix are then used as inputs in the calculation of the evaluation metrics that will evaluate the performance of the implementation, i.e. `precision`, `recall` and `f-measure`. These metrics can be defined as such:

- `Precision` is the fraction of correctly identified events relative to the total number of events identified, and is obtained through the combination of the number of `correct` and `false` results:

$$Precision = \frac{\#\texttt{True Event}}{\#\texttt{True Event} + \#\texttt{False Event}} \tag{5.1}$$

- `Recall` is the fraction of correctly identified events relative to the actual number of existing events. It is obtained through the combination of `correct` and `missing` results:

$$Recall = \frac{\#\texttt{True Event}}{\#\texttt{True Event} + \#\texttt{False No Event}} \tag{5.2}$$

- `F-measure` is the combination of `Precision` and `Recall`:

$$F - measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{5.3}$$

5.3.3 Results

By evaluating the results based on the given methods, seven different confusion matrices were made for each of the different event types. Tables 5.4, 5.5 and 5.6 display the matrices for the `Crime` related events, `Location` related events and for the `Public` and `Ephemerid` events, respectfully.

Crime	Event	No Event	
Event	14	4	18
No Event	11	21	32
	25	25	50

Trial	Event	No Event	
Event	16	0	16
No Event	7	27	34
	23	27	50

Prision	Event	No Event	
Event	9	1	10
No Event	10	30	40
	19	31	50

Table 5.4: Confusion Matrices for Crime related events.

Static	Event	No Event	
Event	3	0	3
No Event	5	42	47
	8	42	50

Visit	Event	No Event	
Event	1	0	1
No Event	17	32	49
	18	32	50

Table 5.5: Confusion Matrices for Location related events.

Public	Event	No Event	
Event	1	2	3
No Event	10	37	47
	11	39	50

Ephemerid	Event	No Event	
Event	19	2	21
No Event	18	11	29
	37	13	50

Table 5.6: Confusion Matrices for Public events and Ephemerids.

With the information obtained from these matrices, it is possible to compare the results, of correctly identified results, with the baseline in Table 5.7.

It is possible to verify that the systems event detection ability increased for all of the event types, with a significant increase in the detection of events related to `Trial` (24%), `Prision` (18%) and `Ephemerid` (29%). The improvements in `Static`, `Visit` and `Public` cannot be considered that relevant

	Crime	Trial	Prision	Static	Visit	Public	Ephemerid
Total	25	23	19	8	18	11	37
Baseline	13	4	0	0	0	0	3
Implementation	14	16	9	3	1	1	19
Improvement (%)	2 %	24%	18%	6%	2%	2%	29%

Table 5.7: Comparison of correctly identified events.

	Precision	Recall	F-measure
Crime	0.78	0.56	0.65
Trial	1.0	0.7	0.82
Prision	0.9	0.47	0.62
Static	1.0	0.38	0.55
Visit	1.0	0.06	0.11
Public	0.33	0.09	0.14
Ephemerid	0.9	0.51	0.66
Total	0.84	0.39	0.51

Table 5.8: Measure results for the event identification.

as the number of recognized events is very low.

The results obtained from the confusion matrices were then compiled in order to generate the results that are displayed in Table 5.8.

For the evaluation of the extracted relations, of the correctly identified events, Tables 5.9, 5.10 and 5.11 display the matrices for the relations of `Crime` related events, `Location` related events and for the `Public` and `Ephemerid` events, respectfully.

Crime	Relation	No Relation	
Relation	14	1	15
No Relation	14	5	19
	28	6	34

Trial	Relation	No Relation	
Relation	38	6	44
No Relation	3	1	4
	41	7	48

Prision	Relation	No Relation	
Relation	3	0	3
No Relation	7	7	14
	10	7	17

Table 5.9: Confusion Matrices for the relations of Crime related events.

The results obtained from these confusion matrices were compiled in order to generate the results that are displayed in Table 5.12.

Static	Relation	No Relation	
Relation	5	1	6
No Relation	6	0	6
	11	1	12

Visit	Relation	No Relation	
Relation	3	0	3
No Relation	0	0	0
	3	0	3

Table 5.10: Confusion Matrices for the relations of Location related events.

Public	Relation	No Relation	
Relation	3	0	3
No Relation	0	0	0
	0	0	3

Ephemerid	Relation	No Relation	
Relation	13	11	24
No Relation	5	1	6
	18	12	30

Table 5.11: Confusion Matrices for relations of Public events and Ephemerids.

	Precision	Recall	F-measure
Crime	0.93	0.5	0.65
Trial	0.86	0.93	0.89
Prision	1.0	0.3	0.46
Static	0.83	0.45	0.59
Visit	1.0	1.0	1.0
Public	1.0	1.0	1.0
Ephemerid	0.54	0.72	0.62
Total	0.88	0.7	0.74

Table 5.12: Measure results for the relation identification.

	Precision	Recall	F-measure
Crime	0.93	0.5	0.65
Trial	0.86	0.93	0.89
Prision	1.0	0.3	0.46
Static	0.83	0.45	0.59
Ephemerid	0.54	0.72	0.62
Total	0.63	0.58	0.64

Table 5.13: Measure results for the relation identification of the selected events.

5.3.4 Discussion

Through the analysis of the Table 5.8, it is possible to discern that the overall *precision* of the implementation is of 84%, which indicates high ability of the system to correctly identifying an event in a sentence without wrongfully identifying that a sentence has an event. However, the high values for *precision* for most of the events can be attributed to the rather low number of sentences that were evaluated. This low number of sentences implied an even lower number of actual events that could be identified for each type which, in turn, does not reflect a proper evaluation of the systems precision.

The *recall* of the system takes a values of 39%, which implies that the system fails at detecting a large portion of events. The values of *recall* are specially low for the events of `Visit` (6%) and `Public` (9%), indicating that they are not detecting most of the events in the sentences. On the other hand, `Trial` has a *recall* of 70%, indicating a very good detection rate of actual events.

Both the *precision* and the *recall* produced an overall *f-measure* of 0.51, revealing an average performance of the system. The individual performances of most of the events were over 0.5 while the events of `Visit` and `Public` had a very low *f-measure* of 0.15, indicating very poor performance.

By analyzing the results of the evaluation of relation extraction in Table 5.12, it is possible to discern very high performance on two event types, i.e. `Visit` and `Public`. This can be attributed to the very low number of events whose relations were evaluated in this phase. In order to better evaluate the performance of the relation extraction of the other events, the results of `Visit` and `Public` will be ignored, in Table 5.13

Now, it is possible to discern that, overall, the *precision* is of 63%. From the considered events, `Trial` had 100% *precision* while having the lowest *recall* of 30%, meaning that, although it failed to extract most of the relations, it correctly identified the ones it did. This resulted on this event having the lowest *f-measure* score of 46%. The `Crime` event, also had a very high *precision* of 93% while failing in extracting half of its relations. The `Trial` event, on the other hand, was able of extracting most of its relations, having a *recall* of 93%, failing just on the correct identification of a few (14%).

This evaluation resulted on a *f-measure* of 64% for the identification of the relations in the correctly

identified events, in these selected events.

Overall, the evaluation of the system was severely influenced by the very low number of testing sets of sentences, and some events, i.e. `Visit`, `Static` and `Public` were, also, affected by the nature of the corpus used, whose context lacked the cases which could be used as evaluating sentences. Another situation that may have lead to failure was the process of extraction of the sentences, in which, the regular expressions used may not have been properly built, which could have lead to a set of less appropriate set of sentences.

6 Conclusions and Future Work

The objective of this work was to develop the capabilities of the STRING chain to identify and classify new event types.

The process of development began with the review of what can be considered to be an event in the context of STRING. Then, the process of reviewing what kinds of information could be considered to be relevant enough to be extracted began. This resulted in a list of well defined relations that can be extracted, for each event type. Some of these relations became core for the event they were part of, which made it so that the event could only be extracted if it was possible to extract that relation.

In total, three different categories of events were developed, i.e. the `crime`, `location` and `public` category.

The `crime` category, was split into three different types of events, i.e. the `crime`, `trial` and `prision` events, of which the `crime` event was considered the core for the whole category, as neither of the other events could be extracted without the prior extraction of the core `crime` event. This category dealt with events associated with the execution of a crime.

The `location` category, was split into two types, i.e. `static` and `visit`, of which, `static` identified events where an entity was in a state of permanence (immobile) and `visit`, which identified events where human entities moved to a location.

The `public` category, was split into two types, i.e. `public` and `ephemerid`, of which, `public` dealt with five different sorts of social events and `ephemerid` dealt with the celebration of the anniversary of important past events.

The development of the system resulted in a more coordinated set of rules which lead to methods of simplification of the methods of writing new rules.

A new set 608 new syntactic rules was created for the expansion of the Statistical and Rule-based natural language processing (STRING) event identification task and, over 450 new entries were added to the pool of existing lexicons.

The evaluation of the system revealed an average f-score measure of 51%, indicating an average performance on event identification while, for relation identification, it yielded a overall f-score measure of 74%, which was influenced by the performance values of two under-evaluated events, which increased

the performance value. The results were partially affected by the combination of a small evaluation set used during the evaluation, errors during the extraction of sentences and the nature of the corpus used, which was unfit for the evaluation of some of the event types developed during this work.

Overall, STRING is a system in continuous development, meaning it is open for future improvements and there are some aspects that can be improved upon and features that can be added. The public event may be further expanded to support more than the current 5 subtypes, with situations related with sports or education. In a similar way, the *efeméride* subtype for the Ephemerid event, can be divided into different subtypes, in an effor to provide differentiation on the events identified by that event subtype. The lexicon pool can be further improved to aid the identification of events and the currently existing events can also be reorganized, which would improve efficiency and facilitate further modification to the rule sets.

Bibliography

Código penal. http://codigopenal.pt/. Accessed: 2018-05-29.

Twitter streaming api. = https://developer.twitter.com/en/docs.

Ait-Mokhtar, S., J. P. Chanod, and C. Roux (2002). Robustness Beyond Shallowness: Incremental deep parsing. *Natural Language Engineering*.

Al-smadi, M. and O. Qawasmeh (2016). Knowledge-based Approach for Event Extraction from Arabic Tweets.

Allan, J. (Ed.) (2002). *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers.

Allan, J., V. Lavrenko, D. Malin, and R. Swan (2000). Detections, Bounds, and Timelines: UMass and TDT-3. *Information Retrieval*.

Althobaiti, M., U. Kruschwitz, and M. Poesio (2014). Aranlp: A java-based library for the processing of arabic text.

Baptista, J., N. Mamede, V. Cabarrão, and J. Dias (2018). Classification Directives for Events and Relations Extraction between Named Entities in Portuguese Texts.

Baptista, J., N. Mamede, C. Hagège, and A. Maurício (2011, September). Time expressions in portuguese guidelines for identification, classification and normalization. Technical report, L2F-Spoken Language Laboratory.

Baptista, J., D. Oliveira, D. Santos, and N. J. Mamede (2011). Classification directives for named entities in Portuguese texts. Technical report, L2F-Spoken Language Laboratory.

Bikei, D. M., S. Miller, R. Schwartz, and R. Weischedel (1991). Nymble : a High-Performance Learning Name-finder.

Cabrita, V. (2014, Novembro). Identificar, Ordenar e Relacionar Eventos. Master's thesis, Instituto Superior Técnico, Universidade de Lisboa.

Carapinha, F. A. G. (2013). Extração Automática de Conteúdos Documentais. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Charikar, M. S. (2002). Similarity estimation techniques from rounding algorithms. *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing - STOC '02*.

Consortium, L. D. (2008). ACE Arabic Annotation Guidelines for Entities. *Facilities*.

da Silva Romão, L. C. (2007, November). Reconhecimento de entidades mencionadas em língua portuguesa: Locais, pessoas, organizações e acontecimentos. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Dang, H. T. (2006). Overview of duc 2006. In *In Proceedings of HLT-NAACL 2006*.

de Almeida Santos, D. T. (2010, November). Extracção de relações entre rntidades mencionadas. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Diniz, C., N. J. Mamede, and J. C. S. D. Pereira (2010). RuDriCo2 - A Faster Disambiguator and Segmentation Modifier.

Forney, D. (1973). The Viterbi Algorithm. *Proceedings of the IEEE*.

Hardy, H., V. Kanchakouskaya, and T. Strzalkowski (2006). Automatic Event Classification Using Surface Text Features. *Proceedings of AAAI06 Workshop on Event Extraction and Synthesis*.

Hogenboom, F., F. Frasincar, U. Kaymak, and F. De Jong (2011). An Overview of Event Extraction from Text. *CEUR Workshop Proceedings*.

Jurafsky, D. and J. Martin (2017). Hidden Markov Models. *Speech and Language Processing* (Chapter 20).

Loureiro, J. M. S. (2007, November). Reconhecimento de entidades mencionadas (obra, valor, relações de parentesco e tempo) e normalização de expressões temporais. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Mamede, N. J., J. Baptista, C. Diniz, and V. Cabarrão (2012, April). STRING: An hybrid statistical and rule-based natural language processing chain for portuguese.

March, O. and T. Baldwin (2008). Automatic Event Reference Identification. *Proceedings of the Australasian Language Technology Association Workshop 2008*.

Marques, J. S. (2013). Anaphora Resolution in Portuguese: an Hybrid Approach. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Maurício, A. S. B. (2011, November). Identificação, Classificação e Normalização de Expressções Temporais. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Oliveira, D. (2010). Extraction and classification of named entities. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa. MSc Dissertation.

Petrović, S., M. Osborne, and V. Lavrenko (2010). Streaming First Story Detection with Application to Twitter. *NAACL HLT 2010 - Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Main Conference* (June).

Pustejovsky, J., B. Ingria, and R. Sauri (2005). The Specification Language TimeML. *The language of time: A reader*.

Ribeiro, R. (2003). Anotação Morfossintáctica Desambiguada do Português. Master's thesis, Instituto Superior Técnico.

Stokes, N., J. Rong, B. Laugher, Y. Li, and L. Cavedon (2007). NICTA's Update and Question-based Summarisation Systems at DUC 2007. *of the Document Understanding Conference Workshop*.

Vicente, A. M. F. (2013, June). LexMan: um Segmentador e Analisador Morfológico com Transdutores. Master's thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa.

Witten, I. H., E. Frank, and M. Hall (2011). *Data Mining: Practical Machine Learning Tools and Techniques*.