UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

ENHANCING EXTRACTIVE SUMMARIZATION WITH AUTOMATIC POST-PROCESSING

Sara Maria da Silveira Botelho da Silveira

Doutoramento em Informática
Especialidade em Ciência da Computação

2015

UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

ENHANCING EXTRACTIVE SUMMARIZATION WITH AUTOMATIC POST-PROCESSING

Sara Maria da Silveira Botelho da Silveira

Tese orientada pelo Prof. Doutor António Manuel Horta Branco, especialmente elaborada para a obtenção do grau de doutor em Informática (especialidade Ciência da Computação)

2015

Abstract

Any solution or device that may help people to optimize their time in doing productive work is of a great help. The steadily increasing amount of information that must be handled by each person everyday, either in their professional tasks or in their personal life, is becoming harder to be processed. By reducing the texts to be handled, automatic text summarization is a very useful procedure that can help to reduce significantly the amount of time people spend in many of their reading tasks.

In the context of handling several texts, dealing with redundancy and focusing on relevant information the major problems to be addressed in automatic multi-document summarization. The most common approach to this task is to build a summary with sentences retrieved from the input texts. This approach is named **extractive summarization**. The main focus of current research on extractive summarization has been algorithm optimization, striving to enhance the selection of content. However, gains related to the increasing of algorithms complexity have not yet been proved, as the summaries remain difficult to be processed by humans in a satisfactory way.

A text built from different documents by extracting sentences from them tends to form a textually fragile sequence of sentences, whose elements tend to be weakly related. In the present work, tasks that modify and relate the summary sentences are combined in a **post-processing procedure**. These tasks include sentence reduction, paragraph creation and insertion of discourse connectives, seeking to improve the textual quality of the final summary to be delivered to human users. Thus, this dissertation addresses automatic text summarization in a different perspective, by exploring the impact of the post-processing of extraction-based summaries in order to build fluent and cohesive texts and improved summaries for human usage.

Keywords: Natural Language Processing, multi-document summarization, sentence reduction, paragraph creation, insertion of discourse connectives, discourse relations, text cohesion, text readability, human evaluation.

Resumo

Qualquer solução ou dispositivo que possa ajudar as pessoas a optimizar o seu tempo, de forma a realizar tarefas produtivas, é uma grande ajuda. A quantidade de informação que cada pessoa tem que manipular, todos os dias, seja no trabalho ou na sua vida pessoal, é difícil de ser processada. Ao comprimir os textos a serem processados, a sumarização automática é uma tarefa muito útil, que pode reduzir significativamente a quantidade de tempo que as pessoas despendem em tarefas de leitura.

Lidar com a redundância e focar na informação relevante num conjunto de textos são os principais objectivos da sumarização automática de vários documentos. A abordagem mais comum para esta tarefa consiste em construir-se o resumo com frases obtidas a partir dos textos originais. Esta abordagem é conhecida como **sumarização extractiva**. O principal foco da investigação mais recente sobre sumarização extrativa é a optimização de algoritmos que visam obter o conteúdo relevante expresso nos textos originais. Porém, os ganhos relacionados com o aumento da complexidade destes algoritmos não foram ainda comprovados, já que os sumários continuam a ser difíceis de ler.

É expectável que um texto, cujas frases foram extraídas de diferentes fontes, forme uma sequência frágil, sobretudo pela falta de interligação dos seus elementos. No contexto deste trabalho, tarefas que modificam e relacionam frases são combinadas num procedimento denominado **pós-processamento**. Estas tarefas incluem a simplificação de frases, a criação de parágrafos e a inserção de conectores de discurso, que juntas procuram melhorar a qualidade do sumário final. Assim, esta dissertação aborda a sumarização automática numa perspectiva diferente, estudando o impacto do pós-processamento de um sumário extractivo, a fim de produzir um texto final fluente e coeso e em vista de se obter uma melhor qualidade textual.

Palavras-chave: Processamento da Linguagem Natural, sumarização multi-documento, simplificação de frases, criação de parágrafos, inserção de conectores, relações de discurso, coesão textual, legibilidade, avaliação humana.

Resumo Alargado

Quando confrontadas com a tarefa de resumir um texto, as pessoas normalmente tentam compreendê-lo de uma perspectiva global. Embora cada indivíduo interprete e transmita a informação à sua maneira, considerando não só a informação veiculada pelo texto mas também o seu conhecimento do mundo, as pessoas constroem um sumário que, normalmente, apresenta as ideias principais do texto original. Apesar disso, indivíduos dife-rentes criam sumários diferentes para o mesmo texto. Assim, ao criar um sumário, são diversas as variáveis em jogo que são difíceis de controlar, mais ainda de reproduzir automaticamente.

A investigação nesta área tem deixado de lado a qualidade textual do sumário final, sobretudo no que diz respeito à utilização que o utilizador humano quer fazer desse sumário. Halliday and Hasan (1976) definem um texto como uma sequência de uma linguagem falada ou escrita que forma um todo unificado, independentemente da sua extensão. Os sistemas actuais de sumarização automática ainda produzem resumos que não são um "todo unificado", e muito menos formam uma sequência fluente. Estes sumários não podem ser verdadeiramente considerados textos, uma vez que formam simplesmente um grupo de frases tipicamente pouco interligadas. Este fenómeno pode ser justificado pelo facto de a maioria destes sistemas não ter sido pensada para ser utilizada por pessoas. Um sistema deste tipo é, normalmente, parte do processamento de um sistema maior. Assim, a principal preocupação tem sido a selecção de conteúdo que fará parte do sumário, enquanto que a articulação do texto final não tem merecido atenção suficiente. Na verdade, os sumários produzidos utilizando este tipo de abordagens são de utilidade limitada para um utilizador humano, uma vez que as tarefas de ler e compreender uma sequência de frases nestas condições não são simples, acabando por não atribuir aos sumários assim produzidos o estatuto de fonte de informação segura.

A **sumarização automática multi-documento** recebe como entrada um conjunto de textos, seleciona as suas partes mais relevantes e devolve um texto mais curto, construído a partir de passagens obtidas nos textos originais ou de texto gerado automaticamente.

Várias são as questões que surgem aquando da construção de um sistema deste tipo. Como selecionar o conteúdo relevante? Como construir um resumo com a taxa de compressão solicitada? Como tratar e organizar todos os temas expressos nos textos originais? Como garantir a coesão do sumário final? Como torná-lo um texto legível?

Com a presente dissertação pretende-se contribuir para a solução de todas estas questões.

O primeiro objetivo é, então, o de construir sumários coesos e de fácil leitura, por meio de um sistema automático. Esses sumários serão criados através da extracção de passagens de uma colecção de textos escritos em Português, definindo-se o sistema como um **sumarizador multi-documento extractivo**.

Nos últimos anos, a principal preocupação no que diz respeito à melhoria dos sistemas de sumarização tem sido a optimização dos algoritmos de selecção de conteúdo, que tendem a ser cada vez mais sofisticados. No entanto, o recurso a estes algoritmos complexos não se reflete em melhorias significativas nos sumários gerados. Após a selecção da informação que compõe o sumário, a forma como esta será organizada no texto final é um desafio que tem sido constantemente relegado para segundo plano, uma vez que o principal objectivo de um sistema deste tipo – a seleção do conteúdo do sumário – já foi realizado. Apesar do algoritmo de selecção de conteúdo ser uma questão importante que não deve ser negligenciada, os sistemas de sumarização automática também devem ter em conta a qualidade do texto final.

Na verdade, considerando a sumarização extractiva, é previsível que os textos produzidos sejam pouco coesos, principalmente devido à natureza do processo extractivo. Kaspersson et al. (2012) estudaram os erros linguísticos mais comuns em sumários extractivos e demonstraram que os erros na extracção afectam a coesão e a fluência global de um sumário. Assim, o pós-

processamento de um texto, após as frases que o compõem terem sido seleccionadas, procura melhorar a coesão dos sumários produzidos.

Assim, esta dissertação aborda a **sumarização extractiva** dando ênfase a um ângulo diferente. O objectivo deste trabalho é adoptar algoritmos simples e eficazes, comummente utilizados, de forma a seleccionar o conteúdo do sumário e concentrar o esforço principal no pós-processamento desse conteúdo, através da criação de frases mais simples e mais bem interligadas, para que o sumário gerado constitua um todo consistente, fluente e legível.

O **procedimento automático de pós-processamento** é o principal enfoque deste trabalho e inclui várias etapas que visam produzir um texto coeso. A hipótese subjacente é a de que é possível construir um sumário com qualidade textual, apesar do seu conteúdo ter sido extraído automaticamente de fontes distintas.

Considerando a colecção de frases que foram obtidas a partir dos textos originais, e depois de assegurada a taxa de compressão, é construída uma versão preliminar do resumo. No entanto, esta primeira versão consiste num texto semelhantes aos produzidos por outros sistemas de sumarização. Posteriormente, são realizadas várias modificações. Primeiro, ao nível das frases e, em seguida, ao nível do texto, de forma a criar um sumário que possa, efectivamente, ser considerado um texto. As frases serão simplificadas e a informação, que pode ser dispensada sem comprometer a compreensão geral do texto, é removida. A ordem do conteúdo do texto é então definida, sendo as frases ordenadas e agrupadas por temas e por parágrafos, para que o texto seja mais fácil de ler por um utilizador. Finalmente, as ligações entre as frases são fortalecidas por meio da inserção de conectores discursivos, que visam apoiar a compreensão do texto.

Em suma, esta dissertação visa contribuir para o avanço da tarefa de sumarização automática através da aplicação de tarefas de pós-processamento a sumários construídos a partir de uma abordagem extractiva de sumarização.

Acknowledgements

(mainly in Portuguese)

Gostaria de agradecer a todos aqueles que, directa ou indirectamente, contribuíram para que a concretização desta Tese fosse uma realidade.

Ao meu orientador, Professor Doutor António Branco, primeiro por ter assumido a orientação deste trabalho; depois pelo incentivo, pela disponibilidade, pela orientação.

Aos meus colegas e amigos do NLX, pelo óptimo ambiente, pela ajuda pronta, muito em especial ao João e à Catarina pela partilha de ideias e por tudo o que fizeram para que esta Tese fosse assim: esta Tese é também um bocadinho vossa; ao Marcus pelas trocas de ideias sempre úteis, numa fase tão necessária; às meninas por todas as risadas.

A todas pessoas que participaram nos testes aqui apresentados; o meu muito obrigada a todos porque sem essa ajuda nada disto seria possível.

À Fundação para a Ciência e Tecnologia (FCT) pelo financiamento desta investigação (SFRH/BD/45133/2008) e ao DI-FCUL pelo apoio logístico.

To everyone in Columbia University, specially Prof. Kathy McKeown, Kapil, Or, Shay, Ioannis, Josh, Karl, Esther. Thank you all for your help and ideas, that definitely enriched my work; and for making my stay in New York even special.

To everyone in iLabs (thanks Ben for being my jquery specialist) and the *Living* team, at Capco, UK, who have always shown so much interest in what took over most of my weekends.

A todos os meus amigos, pela companhia, pelo divertimento, por me terem ajudado a manter a sanidade mental; ao Diogo um autêntico comparsa, não sei o que seria de mim sem as nossas conversas de sonhos acordados; à Sílvia pela partilha de inquietações, de alegrias, e de tudo; à Rita e à Carolina por tanta loucura saudável; à Ana pela amizade de uma vida e porque és uma

fonte de inspiração, também na dedicação a esta vida da investigação; à *stôra* Sónia por ser o exemplo que me fez chegar até aqui.

À minha avó Carolina e à minha mãe por me terem guiado na altura certa. Se não fossem elas, esta Tese nunca seria desta área. A toda a minha família pelo apoio e por todos os momentos de descontração que estar convosco proporciona.

Às minhas irmãs, pela certeza de estarem sempre lá, pela partilha de sempre, pelas loucuras, pelas brincadeiras, pelas turras. Aos meus sobrinhos e aos meus cunhados que sempre me ajudaram a descansar a cabeça e a não pensar no trabalho.

Aos meus pais, por tudo! Pelo amor e apoio incondicionais, especialmente quando cometi a loucura de trocar uma vida estável pela incerteza da investigação; por toda a vossa ajuda, desde sempre; por todos os ensinamentos; pelo grande exemplo que são; por acreditarem sempre em mim.

Finalmente, ao David pela paciência, dedicação, incentivo e inspiração; pela crença inabalável em mim; pelo carinho, amizade e amor incondicionais; por estares sempre presente, em todos os momentos, mesmo que às vezes de longe, essa presença que é um autêntico chão... Sabes bem que se não fosses tu esta Tese nunca teria sido terminada. Muito e muito obrigada!

À minha família: em especial à minha mãe e à minha avó Carolina.
Ao David.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Context and motivation

Human summarization is an interesting and complex task. When addressing the automation of this process, it is clear that this is an even greater challenge. The process a human carries out to summarize a text includes skills and background knowledge that are very hard to model by computational means.

Both Hovy (2004) and Radev et al. (2002) agree when defining a summary. They state that a summary is a text produced from one or more texts containing a significant portion of the information in the original text(s), and that is no longer than half of the original text(s). Automatic text summarization is then the process of creating a summary from one or more input text(s) through a computer program. This is a Natural Language Processing (NLP) research area whose interest has increased significantly in recent times. The information overload phenomenon accelerates the need of creating cohesive and correct summaries by machines, to be easily and rapidly processed by humans.

Keeping the relevant information and compressing the original documents are the key goals of automatic text summarization. Moreover, as the output of such a system aims to be consumed by humans, it must produce a readable text. When compared to related NLP research fields, this is in fact what is distinctive of automatic text summarization (Mani,

2001b). Summarization is thus different from other related text processing tasks: *Text Compression* condenses a text input treating it as code; *Information retrieval* identifies, in a document collection, the documents that are considered relevant to the users' needs; *Information Extraction*, in turn, seeks to fill templates (or tables) using natural language input, which can feed a text generator to produce text output, though not considering compression as a major issue to be taken into account; *Question Answering* systems seek to find, in a collection of documents, the answer for a specific question, posed in natural language; and, finally, *Text Mining* is focused in characterizing singularities on the data, more than condensing it.

Summarization can be relevant to many areas, such as: generation of news headlines, tables of contents of books, TV or cinema digests, email or meeting dialog highlights, abstracts of scientific papers, biographies, obituaries, reviews (of books or CDs or DVDs), etc. Therefore, current applications of automatic summarization include: *news summaries* – e.g. summarize the daily news; *meeting summarization* – e.g. find out what happened at the meeting; *intelligence gathering* – e.g. create a 500-word biography of Barack Obama; *screen-size summaries* to fit in hand-held devices; and *aids for the handicapped* – e.g. compact the text and read it out for a blind person.

Each one of these have different characteristics, such as diverse scope, structure and objectives, which depend on the intentions of the authors. Hence, for the same source text, more than one summary can be produced. This consists of an important challenge to automatic summarization: defining for each situation which is the appropriate summary, while expressing the main idea of the source text.

In order to be useful to all these applications, summaries must be a reduction of the original text(s), while remaining relevant to the goals of the users, and maintaining the informativeness of the text(s).

1.2 Problem

Summarization is a procedure that helps to reduce the text to be read by the user and that can be of a great value in order to optimize not only the time spent in reading topics of interest, but also the space in which the information is stored. Thus, it is essential to take into account the users' needs when building such a system.

Carrying out this task automatically is a non-trivial challenge. Ensuring several prop-

erties, including the preservation of the main idea of the text, selecting the most relevant content of it, removing the redundancy, while ensuring the desired compression rate, as well as obtaining a coherent, cohesive, readable and fluent output summary, is a complex task to be accomplished by a human let alone by a computer.

Examples of summaries are shown below, both in Portuguese and in English. Example 1.1 presents a summary created by the automatic summarization system *GistSumm* (cf. Section 2.1.6), from a source text written in Portuguese (represented in *Annex A.2*), while Example 1.2 shows a summary for the same source text, but created by a human.

Example 1.1: Summary generated automatically (Portuguese).

BNDES destina US$ 1 bi para pequenas empresas
O BNDES (Banco Nacional de Desenvolvimento Econômico e Social) tem disponível US$ 1 bilhão em cinco linhas de financiamento para pequenas empresas industriais, comerciais e de serviços. O BNDES está acertando convênios com entidades de classe, que determinarão produtos e serviços a ser financiados. O POC automático financia investimentos em instalações, infra-estrutura, programas de treinamento, qualidade e produtividade, controle do meio ambiente e desenvolvimento tecnológico. O BNDES automático segue as mesmas regras do POC automático, porém para financiamentos entre US$ 1 milhão e US$ 3 milhões. "O BNDES está destinando US$ 1 bilhão este ano para financiamentos a micro e pequenas empresas", afirma Laércio Gonçalves, 47, chefe do escritório do BNDES em São Paulo. Para ter acesso aos recursos, o empresário deve apresentar no banco de sua preferência um orçamento do produto ou projeto que pretende implantar na empresa.

Example 1.2: Summary created by a human (Portuguese).

BNDES destina US$ 1 bi para pequenas empresas.
Empréstimo deve ser pedido nos bancos; garantias são de 130% do valor.

O BNDES dispõe de US$ 1 bilhão em cinco linhas de financiamento para pequenas empresas industriais, comerciais e de serviços. No final de junho, foi lançada a linha Enter/BNDES, aberta também a profissionais liberais.A finalidade é a compra de equipamentos de computação e programas nacionais e custeia projetos de informatização e treinamento. O BNDES está acertando convênios com entidades de classes, que selecionarão o que deve ser financiado.

Segundo o chefe do escritório do BNDES de São Paulo, Laércio Gonçalves, para alcançar os recursos, o empresário deve apresentar ao seu banco um orçamento do que visa financiar. Este avalia a proposta e solicita outros documentos referentes à empresa. Os prazos para a liberação dos recursos variam de acordo com o tempo gasto para a avaliação preliminar até chegar ao BNDES.

Example 1.3 presents two automatic summaries generated by different summarization systems, and Example 1.4 presents the summary created by a human for the source text illustrated in *Annex A.2*.

Example 1.3: Summaries generated automatically (English).

Summarizer#1: Intellexer Summarizer (available in `http://summarizer.intellexer.com/`)
While Roberts appeared on the cover of the People magazine special edition, other celebrities were not ranked but included Jennifer Aniston, Rihanna, Taylor Swift, Beyonce, Bradley Cooper, Johnny Depp and Patrick Dempsey.
Summarizer#2: SSSummarizer (available in `http://sssummarizer.smartcode.com/`)
Her "Twilight" co-star, heartthrob Robert Pattinson, as well as Canadian teen singer Justin Bieber and 2009 "American Idol" runner-up Adam Lambert made it to the male section of "world's most beautiful", while actresses Jessica Biel, Jessica Alba and singer Jessica Simpson were featured in a new "Beautiful Jessicas" section.

Example 1.4: Summary created by a human (English).

Julia Roberts topped the list of People magazine's "World's Most Beautiful People" on Wednesday, marking the 12th time that the "Pretty Woman" star has appeared in the annual special issue.

When faced with the task of summarizing a text, humans typically start by trying to understand it from a global perspective. Albeit each individual interprets and transmits the information differently, considering both the information conveyed by the text and their background knowledge, humans build a summary that, typically, presents the main ideas of the original text. Nevertheless, different individuals tend to produce different summaries for the same text. Thus, when creating a summary, there are several variables at stake, which are difficult to control, not to mention to reproduce automatically.

Additionally, the textual quality of the output summary regarding the end-user usage has been disregarded, by most of current research on automatic summarization. A text is defined by Halliday and Hasan (1976) as a sequence of a spoken or written language, of any extension, that forms an unified whole. Current summarization systems still produce summaries that do not consist of a "unified whole", much less form a fluent sequence. Therefore, according to the definition of text stated above, these summaries cannot be considered as texts since they simply form a sequence of sentences weakly connected. A reason that may justify this lies in the fact that most of these systems have not been designed to present their output to a human user. They are rather embedded in a larger system. For instance, in order to perform document clustering, documents are first automatically summarized to reduce the search space. In fact, the main issue addressed has been the selection of the content which will be part of the summary, while the articulation of the final text has been given much less or no attention. So, summaries produced using this kind of approach can hardly be useful to a human user, since the tasks of reading and

comprehending the summary are not straightforward and, rather than being helpful, the summaries produced do not constitute a valuable source of information.

In short, extracting sentences using increasingly sophisticated techniques has been the issue of concern in the literature in the last years, while textual quality has been mostly disregarded so far. Still, those techniques are bringing little improvements, if any, as it is not in their nature to cope with this problem. Textual quality of extraction-based texts is thus the focus of our work, as there is much room for progress.

1.3 Goals

Automatic text summarization receives a text (or a set of texts) as input, builds a representation of the text(s) content, selects the most relevant parts, that reflect the text(s) purpose, and returns a smaller text, that consists either of parts retrieved from the original text(s) or by newly automatically generated text.

Many issues arise while building such a system. How to select the relevant content? How to build the summary with the requested compression rate? How to address and organize all the topics in the documents? How to ensure output summary cohesion? How to make it a readable text?

These are the kind of questions that this dissertation aims to address. The very first goal of this work is **to build summaries with textual quality for human users, through an automatic system**. These summaries will be built by extracting passages from a collection of texts, designing the automatic summarization system as an extractive and multi-document one. While, in this dissertation, the working language of the text resorted to Portuguese, the solutions that will be sought are, in their essence, language independent, and can thus be applied to other languages.

One of the major advantages of computers is their ability to process huge amounts of data in very short time. There are tasks that if executed manually would be discarded at the inception for the time and effort they would incur. Yet, if performed using a computer, their execution becomes feasible. This is the case of the task that aims to build summaries from large collections of texts, the multi-document summarization task. Processing huge amounts of data within texts is an easy task for a computer, yet decipher which of that disperse and unstructured data is relevant is a great challenge. Furthermore, this data is expected to contain not only redundant information, but also possibly inconsistent one,

aspects that increase the complexity of salient information gathering within a collection of texts.

As mentioned above, in recent years, the main research concern when seeking to improve summarization systems has been the optimization of content selection algorithms. Algorithms tend to be increasingly sophisticated, using increasingly complex statistical functions, and aiming at fine-tuning each component of the system in order to improve the system evaluation, usually performed automatically. Nevertheless, these complex algorithms tend to hardly bring relevant improvements for human users. Despite the content selection algorithm being an important issue that should not be neglected, automatic summarization systems must also be concerned with their final output. Once the information that composes the summary has been selected, the way it will be organized in the final text is another challenge that has been disregarded.

This dissertation aims to address summarization from a different perspective. The objective of this work is to build simple and commonly used algorithms to effectively select the summary content, while focusing the effort in the post-processing of this content, by creating simpler sentences and by interconnecting them in order to form a consistent whole, that is a more fluent and readable summary.

Extracting sentences for the summary is an issue that has been extensively researched in the literature in the last decades. Not only the increasingly more sophisticated algorithms for extraction are bringing now little improvements, if any, as it is definitely not in their nature to cope with the problem of textual quality. In our view, improving the quality of the texts extracted is where improvement is more needed given the state of the art. Therefore, another goal to be pursued in this work is thus **to study the possibility of improving the state of the art by incorporating, in an automatic summarization system, a summary post-processing module** that simplifies and articulates the text automatically, after an automatic selection of the sentences that compose the summary.

In fact, considering summarization built by extraction, it is predictable that output texts would have lack of cohesion, due to the nature of the extraction process. Kaspersson et al. (2012) studied the most common linguistic errors in summaries built by extraction. Firstly, they have related the compression rate to the lack of information in the summary. They have shown that the smaller the summaries are, the higher is the probability they have a lack of cohesion and a lack of content, so that the input may end up misrepresented by the summary output. Moreover, they have shown that there are seve-

ral extraction errors that affect the cohesion and the global context of a summary. These are, in fact, expected results that have been shown by this work.

The automatic post-processing procedure is the main challenge of the present work and includes several steps that aim to produce a summary with more textual quality. Our basic assumption is that it is possible to build a summary with more textual quality, even though its content has been extracted from different texts.

In sum, this work aims **to study the impact of post-processing summarized texts to improve their textual quality**.

1.4 Hypotheses

While studying the types of errors in extraction based summaries, Kaspersson et al. (2012) concluded that "absent cohesion or context" errors were the most common ones. Accordingly, the main objective of post-processing an extraction-based summary is to produce a text which forms a cohesive and fluent whole, improving the summary textual quality. A cohesive text is characterized by a consistent association between its sentences, that can be achieved for instance by using connectives that link the sentences together, whereas a fluent text is defined as a simple and readable one that must not be reread to be understood.

The post-processing of a text includes several tasks, namely:

- sentence reduction – produces simpler and more incisive sentences;
- paragraph creation – groups related sentences into topic paragraphs;
- connective insertion – includes conjunctions or adverbs between related sentences.

Consequently, the present work seeks to pursue the following hypotheses:

Hypothesis#1:

Automatic summarization can be improved by reducing sentences while allowing for the reduction of redundancy and maintaining summary informativeness.

Hypothesis#2:

Automatic summarization can be improved by arranging sentences in paragraphs.

Hypothesis#3:

Automatic summarization can be improved by inserting discourse connectives.

Hypothesis#4:

The automatic post-processing of a summary built by extraction improves the textual quality (cohesion, fluency, readability, etc.) of a summary.

Thus, this work aims to produce summaries for **human readers**. In this context, we seek to ensure the textual quality of the generated summary.

1.5 Solution proposed

Many approaches have been proposed in the literature to perform automatic text summarization. Some reuse portions of the input texts, while others automatically generate the content of the summary. In this work, the summaries are composed by content extracted directly from the input texts. However, such approaches typically produce summaries whose sentences are weakly related.

In order to tackle this problem, several experiments with language experts were performed to assess the impact of editing the summary, as a means not only to improve those connections, but mainly to enhance the textual quality of the summaries (Silveira and Branco, 2012). The goal of these experiments is twofold. On one hand, they were a first assessment of our hypotheses. On the other hand, they pointed out which sort of editing operations would be the most effective in improving extract summaries.

The first experiment aimed to understand if the task of smoothing texts by inserting connectives (adverbs, conjunctions, etc.) would enhance automatic summaries. In addition, this experiment sought to determine if changing the sentence order improves the text readability, cohesion and fluency, thus its textual quality. Subsequently, another experiment compared summaries whose sentences were reduced to their main content with the input texts from which these summaries were built. This experiment was performed to investigate if reduced texts preserve the original text meaning, besides being readable, fluent and cohesive.

Initial lessons learned from these experimental explorations lead to pursue the solution proposed in the present dissertation, which combines several editing procedures with state-of-the-art summarizing techniques (Nenkova and McKeown, 2012).

Figure 1.1 sketches the solution pursued in this work.

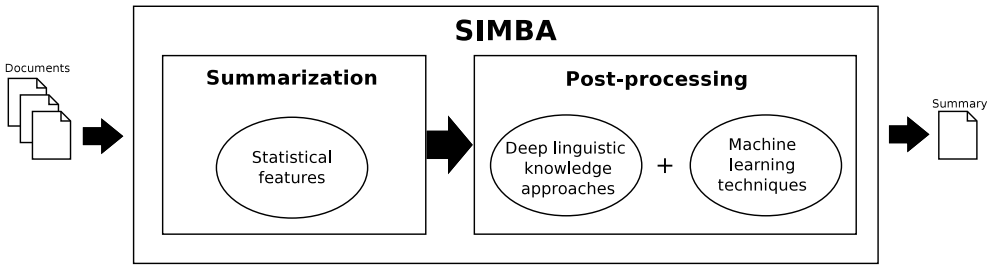


Figure 1.1: Solution proposed.

The solution worked out in this work uses an hybrid approach. Firstly, a statistical algorithm that extracts sentences from the input texts is used to ease and accelerate the core processes of a multi-document summarization procedure: selection of relevant content, redundancy removal and text ordering. Hence, an agile method identifies, in a generic way, the most relevant information that should be included in the summary.

Subsequently, a post-processing module modifies the text, aiming to build a more fluent and readable summary, in order to be consumed by a human user. This module includes a set of tasks that make use not only of deep linguistic knowledge approaches, but also of machine learning techniques. Editing tasks will be performed, first at sentence-level and then at text-level, to produce a better summary. Sentences will be simplified and possibly less crucial information, which is less relevant to the overall comprehension of the text, will be removed. Defining the retrieved content order is another challenge that is addressed. Sentences will be ordered and grouped by topics and respective paragraphs, for the text to be easier to read by a human user. Finally, the connections between the sentences will be strengthen through the insertion of expressions that seek to improve the text comprehension. These expressions are identified based on the discourse relation found between the sentences already selected to be in the summary. This is actually the topic for *CoNLL Shared Task for 2015*[1], in which the systems in competition are required to return the discourse connectives representing the discourse relations found in texts.

Along with a study about automatic summarization, in general, this work contributes with a concrete multi-document summarization system, SIMBA, that produces summaries, from collections of texts. The ultimate goal of the solution proposed is thus to improve the textual quality of the final text, in order to make it useful for human users.

[1] `http://www.cs.brandeis.edu/~clp/conll15st/`

1.6 Dissertation structure

The remaining of this dissertation is organized as follows.

Chapter 2: Background. This chapter focus on the fundamental concepts and relevant background work on the specific research areas addressed in this dissertation. It is divided in three sections. The first section introduces the main research area, automatic text summarization. The second one describes the research done in sentence reduction, either it has or has not been applied to summarization. Finally, in the third section, research done on implicit discourse relations is described.

Chapter 3: Summarization. This chapter describes the multi-document summarization system, SIMBA, that aims to support the goals and the hypotheses previously stated in this chapter. It contains three sections that describe in detail the three main phases of a summarization system: analysis, content selection, and summary generation.

Chapter 4: Post-processing. This chapter details a post-processing module that aims to improve the final summary text that is delivered to the user. This module is part of the summarization system. It is composed by two tasks: compression tasks, that include sentence reduction and compression review, and fluency tasks, composed by paragraph creation, and connective insertion. This module seeks to transform the relevant information gathered during the summarization process into a fluent and readable summary.

Chapter 5: Evaluation. This chapter discusses the evaluation of the solution proposed. Evaluation is divided in two main steps: automatic evaluation and human evaluation. Automatic evaluation uses automatic metrics to determine the informativeness of the generated summaries. Human evaluation, in turn, was performed by asking human users their judgement about the quality of the automatically built summaries.

Chapter 6: Conclusions. This chapter concludes the dissertation, and points out some future research directions.

CHAPTER 2

BACKGROUND

This chapter introduces the research performed in the three major areas more closely related to the work presented in this dissertation. Definitions and methods concerning *text summarization* are introduced in Section 2.1. Section 2.2 reports on approaches for *sentence reduction*. Finally, in Section 2.3, previous work on finding *implicit discourse relations* is presented.

2.1 Text summarization

The first work on automatic summarization dates from the late 1950s. The results obtained were based on statistical techniques, and were applied to summarize scientific texts. Back in the 1970s, the research was focused on domain-based systems, and later on, in the 1980s, artificial intelligence approaches became the basis of automatic summarization, applied mostly over short texts, narratives, and news as input. In the 1990s, hybrid systems, which combined statistical and knowledge-based methods, were used on news and scientific texts. Finally, in the 2000s, generation of headlines, multi-document summarization, along with the requirements induced by hand-held devices to produce smaller texts led to advances in automatic summarization.

Firstly, this section introduces the main notions that must be taken into account in

what concerns automatic text summarization. Afterwards, the three main approaches used to perform summarization – extraction, abstraction and *corpus*-based – are described. Then, evaluation methods are discussed. Finally, some currently available summarization systems are described.

2.1.1 Definitions

This section discusses the main notions to be taken into account when building an automatic summary, the summary features that should be considered, the approaches that can be used and, finally, the structure of the summarization process.

Summary design features

There are many factors that influence the way a text is summarized. Jones (2007) defines three types of features (described in Table 2.1).

Input	**Purpose**	**Output**
language	audience	length
span	function	relation to source
genre		

Table 2.1: Summary features

Input features characterize the source material style and units; the purpose features include the intended use and audience; and the output features address the reduction level and the format of the summary. These features determine not only the way the summarization system processes the input text(s), but also the creation of the system deliverables, taking into account the purpose of the summaries.

Input features

The input features determine the strategy to be adopted by the summarization system, based on the input material.

Regarding **language**, summarization systems can be classified as monolingual – the input language is the same as the output language – or cross-lingual – processes several languages, and the output language differs from the input language(s). Due to the fact

that a cross-lingual system must involve some kind of machine translation, most of the work done in developing summarization systems has been focused on monolingual ones.

Saggion (2006) notes that "the goal of any form of summarization is to take an information source [a text or a collection of texts], extract the most important content from it and present it to the user in a condensed form and in a manner sensitive to the user's needs". Both single- and multi-document summarization aim to build a summary that satisfies these constraints. The difference between these two forms of summarizing lies in the input documents. The system **span** depends on whether the user submits a single document – single-document summarization (SDS) – or a collection of documents – multi-document summarization (MDS).

Single-document summarization aims to retrieve the salient information, that may be disperse in the input document. As an extension to this, multi-document summarization poses new challenges, since the simple concatenation of single-document summaries is not a satisfactory solution. Redundancy and possibly inconsistent information within the collection of documents must be addressed. In addition, the inevitable thematic diversity within a large set of documents is another difficulty to be overcome, as the document collection may share or not a central topic.

Ensuring the summary cohesion is a great challenge specially when the material comes from different sources. An ideal automatic summarization system not only should shorten the input texts, but also it should present the information organized around the key aspects of the collection, in order to represent a wider diversity of views on the topic.

Finally, **genre** defines the type of text which is submitted to the system. Texts can be of many varieties, such as scientific or technical reports, news stories, email messages, editorials, books, etc., and the summarization system may use special strategies to produce the output summary, by taking into account the type of text submitted.

Purpose features

The purpose features are determined by the intended use of the summary.

Concerning the **audience**, a summarization system has several classifications. Generic summarization targets the most salient content of the document(s) and aims at a broad readership community. User-focused summarization adjusts the output summary to the requirements of a particular user or group of users, taking into account the interests and

background of the user. Query-driven summarization is guided by a specific query submitted by the user, expressing his information needs. The output summary must then contain information that answers the query. Topic-oriented summarization involves a topic, which is not expressed in the form of a well-formed question or query, but for instance by a keyword or a collection of keywords. Its output summary must concentrate on a specific topic of interest, containing information relevant to this topic.

Regarding its **function**, a summary can be classified as indicative, informative or critical (Mani, 2001a). Indicative summaries provide an idea of what the text is about. This kind of summaries simply report on the source text content, by identifying its main topic. They provide a reference for selecting documents for more in-depth reading. Informative summaries, in turn, provide some shortened version of the content and, therefore, can replace the original text. These summaries cover all the salient information in the source document, at some level of detail. Finally, critical summaries express different points of view of the subject matter of the text. These texts commonly involve opinions, feedback, identification of weaknesses, recommendations, etc., beyond what is found in the source, expressing also some sort of criticism about the quality of the work of the author. Given the challenges involved, they have been somewhat out of the scope of current automatic summarization systems.

Output features

Depending on the output options, the strategy of the summarization system varies. These features must then be previously defined in order to establish the approach to be pursued.

The summary **length** is determined by its compression rate. The compression rate is given by the ratio of summary length over the source length. Some authors use the sentence as the unit to compute the compression rate, but the most common approach uses words to compute it, since, in automatic summarization, words are considered the smallest units of the text.

Many authors define a high vs. low compression rate. In the present work, we assume 99% as a high compression rate, since the output summary has only 1% length of the original text; and, consequently, a 1% compression rate as a low compression rate as the summary maintains 99% of the original text. The compression rate is commonly set up by the user (or is defined by default, e.g. 70%), determining the summary length.

Several factors may influence the choice of the compression rate. The text genre is one of these factors. For instance, a book may require a higher compression rate than a scientific paper. The summary function is another factor, as by definition indicative summaries demand higher compression rates. Finally, the needs of the users and the space available may also influence the choice of the summary length.

By setting the **relation of the summary to the source**, the summarization process receives yet another characterization. The fundamental distinction lies in the definition of extracts vs. abstracts.

An extract is a summary consisting of material copied from the source (Mani, 2001a). The summary is created by reusing stretches (words, sentences, etc.) of the input text. The extracted stretches may be contiguous or not. An extract may not be composed of complete sentences; it can consist of a list of terms, phrases, truncated sentences, etc. Generally, the process of building an extract, named *Extraction*, may not require any specific knowledge concerning the language or the text genre, being thus the most common approach used in automatic text summarization.

On the other hand, an abstract is a summary whose material – at least some – is not present in the input. Such a summary is created by regenerating the extracted content (Radev et al., 2002). Typically, an abstract contains some degree of paraphrase of the input content. In order to do this, some sort of semantic representation of the input text is required. Moreover, a representation of the output text is also required, in order to build new sentences to form the summary. This method is called *Abstraction*. Also, abstraction may involve inference about the content of the text, or background references, that is concepts that are not explicitly present in the text.

As discussed in Chapter 1, most of nowadays research on automatic summarization is based on extraction, since it is a relatively straightforward solution, compared to the abstraction method, which requires more sophisticated approaches, that involve deeper processing tools, and specific language resources, that are still not available for most of the languages.

Approaches

The approach adopted for summarization determines the methods used to implement a summarization system. The most common approaches can be defined as shallow or

deep. Shallow approaches use statistical and *corpus*-based methods to perform summarization, while deep ones tend to rely more in formal and linguistic theories. These techniques can be combined resulting in a hybrid approach, seeking to explore the best strategies from each methodology in order to optimize the system.

Shallow approaches typically do not go beyond the syntactic level of representation. These strategies produce extracts, usually by extracting sentences, which contain the parts of the source text(s) containing the most salient information. These parts are then arranged and presented in some effective manner. This is a very robust technique since it is mainly domain independent and it does not require the complex resources that a deeper one demands. However, the compression rate achievement is a challenging issue, when this approach is used. Extracted sentences should be added in full to the summary – that is, should not be pruned –, in order for the sentences to remain coherent. But, by adding the complete sentences to the output summary its compression rate may be compromised, since the number of words that should form the summary can be exceeded. This lack of control over the compression rate is the main disadvantage of this approach.

Corpus-based sentence extraction is another shallow method used. This is a technique that compares hand-crafted summaries with their source texts, in order to infer which are the most important features in the text that must be taken into account to produce summaries. It typically uses machine learning techniques that, based on those features, will learn which of the original content should be added to the summary. However, this method is a costly one, as summaries built by professional experts are needed. Once the system is tuned, it can be very effective in summarizing texts from specific domains, with restricted vocabulary. Nevertheless, this is not the most common method used, as one of the most common requirements of summarization systems is the generic use of the produced texts. *Corpus*-based methods are further discussed in Section 2.1.3.

In what concerns shallow approaches, extraction is a relatively straightforward solution with the bulk of the effort directed to the production of extracts. Extraction methods are further presented in detail in Section 2.1.2.

Deeper approaches usually assume at least a semantic representation, at sentential level. They generate abstracts, thus requiring domain information, and can involve natural language generation (Mani and Maybury, 2001). Since the output is generated, the building of a coherent text is sought, by enforcing various constraints, which define the connections between the elements of the semantic representation.

These approaches tend to require some structured or preprocessed input, which is also built by humans, and turns out to be very expensive. Moreover, domain specific knowledge is usually needed, as general-purpose knowledge bases are not available or feasible. Despite the processing complexity and the domain restriction, deep approaches promise more informative and more coherent summaries. In addition, desired compression levels can be achieved, since the text is generated automatically and its passages can be selected considering the length of the text to be created.

Moreover, deep approaches favors abstraction. Typically, these approaches rely on external tools such as parsers, ontologies, or thesaurus, that can provide additional information, based on units of the original text, that might be added to the final summary in order to improve the content of the text. Abstraction methods will be further discussed in Section 2.1.4.

Current work based on abstraction methods is still in an early stage of development since the tools used need to be more accurate. Besides, some of these systems rely on an initial preprocessing phase, which builds training datasets based on templates that specify the information that should be expressed in the summary. On one hand, the creation of the templates is very time consuming. On the other hand, the dataset construction restricts the system domain – as usually the texts used to train belong to a particular topic. This approach is a very costly one, specially considering that its most important promise, the gains related to the compression, has not yet been proven.

Stages

The execution of the summarization process depends on the system span. That is, the strategy adopted to build the summary is distinct whether the input is a single text or a collection of texts (Mani, 2001a).

Considering a single-document summarization system, the most typical summarization process comprises three stages: analysis, transformation and synthesis (Figure 2.1).

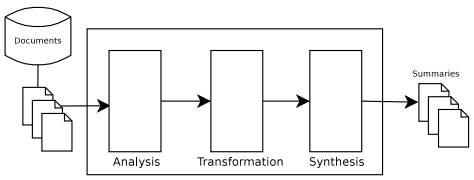


Figure 2.1: High-level architecture for a SDS system (Mani, 2001a).

These stages are applied in sequence, so that the output of one phase is the input of the following phase. The analysis phase aims to analyze the input and to build an internal representation of it. The transformation phase handles this internal representation of the input and transforms it in a representation of the summary. However, it is worth noting that, in some systems, Analysis and Transformation are merged into one single stage that delivers the summary content directly to the Synthesis phase. The synthesis phase renders the summary representation into a text.

On the other hand, considering a multi-document summarization system, the generic algorithm is more granular (Figure 2.2). The previously mentioned three steps enclose five steps. Analysis maps to Identification, while Transformation includes both Matching and Filtering phases. Finally, Synthesis is composed by Reduction and Presentation.

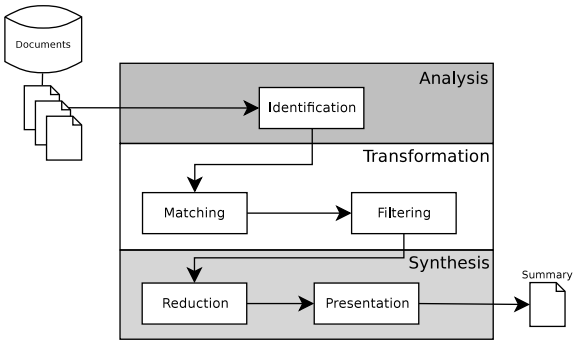


Figure 2.2: High-level architecture for a MDS system.

The identification step recognizes text elements to be obtained from the collection. Then, the instances of these elements across the texts are matched, using an appropriate matching metric, based for instance on notions of string-identity, informational equivalence or informational subsumption. Filtering the matched elements, and selecting salient ones based on some salience criterion, is the next step of the procedure. After this, the matched elements are reduced using aggregation and generalization operations which deliver more succinct elements. Finally, the resulting elements are presented using generation and/or visualization methods (Mani, 2001a).

These high-level architectures should not be taken as a rigid stipulation of how to build a summarization system, but rather they result from the observation of most of the summarization systems algorithms, which have been built according to these processes.

2.1.2 Extraction methods

The very first approaches to automatic text summarization started by performing extraction, using statistical methods that computed word frequencies. However, the summaries produced were incoherent and hard to read, since they were used to feed other systems, thus not being built to be consumed by human users. Despite this, extraction techniques still produce very acceptable results, being thus widely used.

Extraction methods commonly follow the typical summarization process, performed in three stages: analysis, transformation and synthesis.

The **analysis** phase processes the input text(s) and builds an internal representation of those text(s). The approach carried out to perform this analysis phase depends on the summarization system goals. There are typical steps executed by most of the systems, which include tokenization, part-of-speech annotation, lemmatization, or stop-words removal. Also, in this phase, the information units – that can either be sentences or paragraphs – are identified to be handled in the next phases. Henceforth, the information units will be referred to as sentences, since those are the most commonly used.

Once the sentences have been identified, some proposed systems define formal representations of the text taking into account those sentences retrieved. Formal representations include matrices, graphs or clusters, that encode relationships between sentences. On the other hand, other studies score the sentences based on their features. Either way, the main goal is to support the relevance of those sentences. This is the task of the subsequent phase in the processing chain. The **transformation** phase aims to handle the sentences and to identify the ones that are the most relevant within the input text(s). Each sentence is assigned a score. This score can be obtained in different ways, whether some sort of semantic or formal representation of the input text(s) is produced or not. If a formal representation is built, the sentence score is computed on the basis of that representation. Otherwise, measures of significance of the sentence are calculated over the text(s) to infer each sentence score.

A **formal representation** of the input text(s) typically represents the relationships between the sentences using a matrix, a graph or a set of clusters.

A very common way to represent documents is to use a matrix. Document-term (or sentence-term) matrices describe the frequency of terms (words) that occur in a collection of documents (or sentences). Rows and columns correspond to documents (or sen-

tences) and terms, respectively. Each value in the matrix can be computed using similarity metrics, like the cosine similarity (described below), or using weighting schemes, such as `tf-idf` (described below).

Due to the sparse nature of the matrix, once it has been built, the next step lies in its factorization. The most common method used is Singular Value Decomposition (SVD), an algorithm that factorizes matrices into a series of linear approximations that expose the underlying structure of the matrix. Thus, SVD finds a reduced dimensional representation of a matrix that emphasizes the strongest relationships while removing the noise. This way it is possible to identify the most relevant concepts or terms within the collection of documents.

Wang et al. (2008) built a semantic similarity matrix. Pairwise sentence semantic similarity is calculated based on both a semantic role analysis and word relation discovery using WordNet (Fellbaum, 1998). The symmetric matrix factorization is performed to group sentences into clusters. Afterwards, in each cluster, sentences are ranked based on the sentence score, which combines two metrics: (1) the average similarity between the sentence and all the other sentences; and (2) the similarity between the sentence and the given topic.

Also, graphs can be used to represent the relations between text(s) or sentences. Typically, each sentence denotes a node, and the edges, which are labeled, define the relation between the nodes they connect. This relation is commonly computed using a distance function, such as the Euclidean distance or the Hamming distance. The lower the distance between two elements is, the higher is the probability to link them. Once the graph has been built, graph algorithms can be used to compute the best path, that is the path between the sentences that maximizes the edges values. Commonly used algorithms are the ones used in graphs theory, such as depth-first search or breadth-first search. The sentences in the path are the candidates to be integrated in the summary.

Mani and Bloedorn (1999), for instance, represent text items (such as words, phrases, and proper names) in a graph. The nodes in the graph represent word instances at different positions, with phrases and names being formed out of words. Additionally, cohesion relationships – such as synonym/hypernymy, repetition, adjacency, and co-reference – between term instances are explored to determine salient terms. These terms are then used to identify the salient content to be extracted from the input text.

The gathering of matrices or graphs are usually a step previous to clustering, a very

common method used in Information Retrieval to group similar documents. This method has been applied to summarization in order to identify relationships between documents. Clustering algorithms have been extensively tested and can be divided in two main approaches: agglomerative clustering (bottom-up approach), and divisive clustering (top-down approach). Agglomerative algorithms start with each element as a separate cluster and merge them into successively larger clusters. Divisive algorithms, in turn, start with the whole set and proceed to divide it into successively smaller clusters. Dividing or merging elements in clusters requires a similarity measure that determines whether an element should be added to or removed from a cluster. This similarity measure is typically computed using some distance function, as the ones mentioned above.

Radev et al. (2000), for instance, use clustering to group all the documents into clusters of related topics. They use a modified *tf-idf* score to produce clusters of news articles on the same event. Clusters consist of chronologically ordered news texts, which describe an event as it develops over time. Each cluster is represented by a centroid, a document containing the words occurring in the documents of the cluster, which score is above a specified threshold. Afterwards, the sentences are ranked based on a combination of three features: centroid value, positional value, and overlap value. The centroid value is the sum of the centroid values of the words occurring in the sentence. The more centroid words the sentence has, the highest this value is. The positional value refers to sentence position in the document, and it is computed based on the first sentence which gets the same score as the highest-ranked sentence in the document according to the centroid value. Hence, this value decreases linearly as the sentence gets farther from the beginning of a document. Finally, the overlap value is the inner product of the *tf-idf*-weighted vector representations of a given sentence and the first sentence in the document. The final score is a linear combination of these three features, along with the subtraction of a redundancy penalty, that considers the overlap between a given sentence and sentences with higher score values. The summary is then created by selecting the sentences with the highest scores, until the compression rate is met.

When the input text(s) are **not formally represented**, each sentence score is defined using reward and penalty metrics. Typically, these metrics take into account features related to the sentence or the words composing it. The reward ones increase the scores of the sentences, while penalty metrics decrement such scores.

Luhn (1958) proposed the keyword method, which states that more frequent words

in a document indicate the topic discussed, so that sentences containing more frequent words are assigned with reward values. Edmundson (1969) experimented several other methods, which can either be used to reward or penalize sentences. The cue method states that sentences that contain cue words (such as "significant", "impossible", "hardly", etc.) are considered relevant. The location method defines that important sentences occur in specific positions of the text or specific sections in the document. The title method states that the words in the title indicate the content of the text.

As a result, a number of problems were often reported in the summaries built using these methods. These types of problems included lack of balance, lack of cohesion and lack of fluency (Paice, 1990). The methods mentioned above inspired the development of new ways to compute sentence scores, that consist either in the refinement of previous ones, or in the combination of several of them.

The Optimal Position Policy (OPP) (Lin and Hovy, 1997) is a list of positions in the text in which salient sentences are likely to occur. Thus, sentences in these positions are considered the most relevant.

Lin and Hovy (2000) defined topic signatures as sets of related words that can be used to identify the presence of a complex concept, i.e. a concept that consists of several related components in fixed relations. This concept proved to be useful in finding the sentences that convey the most relevant information.

In fact, the most used score measure is *tf-idf* (Term Frequency – Inverse Document Frequency), which is a term weighting measure where the salience of a term in a document is related to the number of documents in which the term occurs. A very frequent term in a collection of documents may not necessarily convey relevant information. Thus, *tf-idf* assigns more relevance to a frequent term in a document that rarely occurs in the whole collection of documents. Sentences containing terms with high *tf-idf* scores are more likely to be significant in the collection of texts. The sentence score is then the sum of the *tf-idf* scores of its words. Also, a new term weighting measure based on *tf-idf* was introduced. Instead of documents, *tf-isf* (Term Frequency – Inverse Sentence Frequency) relates term frequency with sentences. Neto et al. (2000) used *tf-isf* similarly to *tf-idf*, in order to rank terms and to define sentence scores.

Vocabulary overlap measures are also used to define sentence scores. The Dice coefficient (Dice, 1945), the Jaccard index (Jaccard, 1908), and the Cosine similarity coefficient compute a similarity metric between pairs of sentences, determining a relation between

them. For instance, along with other features, and to define the sentence score, Radev et al. (2000) compute the overlap between each sentence and the first sentence of the text, which, in news articles, is considered one of the most important sentences in the text. White et al. (2001), in turn, compute the overlap between each sentence and the document title. The sentences that have a high degree of similarity with the title are considered to be important to the global context.

Other approaches rely on a query or question to drive sentence selection. Mori et al. (2005) defined the sentence score using a Question Answering (QA) system (Mori, 2005), that weights sentences considering their relevance to a specific query. This weight is the sentence score. The sentence score is a linear combination of four matching scores, that relate three elements: the answer candidate – a potential answer to the question submitted –, the current sentence and the query. These matching scores consist of: (1) 2-grams matching; (2) keywords matching; (3) dependency relations between the answer candidate and the keywords; and (4) query type matching. The final score measures the relevance of the sentence in a collection of documents considering the query submitted. This score is the one used as the sentence score.

Farzindar et al. (2005) perform question-driven summarization. Along with other measures, such as cosine similarity and sentence position, they compute the number of named entities in the sentence which have the same category as the named entities in the question. In addition, sentences containing cue-phrases, such as "as consequence", "as a result", "as far as", etc., are rewarded since the sentences containing these phrases may introduce relevant information. The sentence score is a linear combination of several measures.

The metrics already discussed are both suitable for single- or multi-document summarization. However, multi-document summarization brought new challenges. Beyond selecting the most salient information which represents the collection of documents, removing the redundancy present in the texts is another challenging issue to be overcome. Consequently, the improvement of sentence weighting mechanisms to perform a more accurate information selection is thus required.

Many works addressed these challenges by trying new metrics for word or sentence significance. The Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998) is a linear combination metric that relates query-relevance with information-novelty. A document has a high marginal relevance if it is both relevant to the query and contains

minimal similarity to previously selected documents. Thus, the low degree of similarity between a document and the other documents in the collection is linearly combined with the high degree of similarity between that document and the query. This score reflects the importance of the document in the global collection: the highest the score, the more important the document. This criterion strives to reduce redundancy, while maintaining query relevance in ranking retrieved documents and in selecting appropriate passages for text summarization.

Other studies used refined statistical models in their summarization systems. The Latent Dirichlet Allocation (LDA) model (Blei et al., 2003) was the very first probabilistic model applied to summarization.

In their work, Arora and Ravindran (2008) used LDA to split the collection of documents into topics. Each document is represented as a mixture of topics with a probability distribution that represents the importance of the topic for that document. Considering all sentences in the documents and all the topics, the probability P of each sentence S given each topic T, that is $P(S|T)$, is computed. As a sentence consists of a set of words, the more words of the sentence that belong to a specific topic, the higher the probability of the sentence to belong to that topic. So, each topic is represented as a mixture of sentences with a probability. The sentence probability is the sentence score. Thus, the sentences with higher probability from the topics with higher probability are more suited to be selected to compose the summary. This approach is purely statistical and the algorithm does not involve the structure of the documents or the structure of the sentences in terms of grammar or meanings conveyed by the words.

Applying LDA to summarization can be useful in identifying the most relevant topics (and consequently sentences) within a collection of documents. However, a major flaw of this model is the lack of order of these topics, i.e. LDA does not relate topics among each other, and thus it is not possible to infer the order of importance of the topics. In order to address this lack of order among topics, Blei et al. (2004) proposed a hierarchical LDA (hLDA) model, an extension to LDA model, which aims to determine hierarchical relations among topics. First, LDA was used to split the documents into topics. Then, hLDA was used to determine a hierarchy between those topics. hLDA introduces a new level to the LDA process, which aims to represent the topics using a tree that defines the hierarchy of the topics. The highest levels are more general and the lowest levels are more specific.

In combination with reward metrics, **penalty metrics** seek to refine the score of the sentences. Schiffman et al. (2002) propose several penalty metrics. Sentences appearing at the end of a document tend to be less relevant, thus they should be penalized. Also, the ones with less than fifteen words are considered too small and conveying limited information, thus a predefined score value is decremented from the sentence score. Moreover, sentences that start with pronouns are penalized since they may introduce a lack of information in the summary. Finally, they assume that the summaries must contain up-to-date information so that if the text publication date is not a recent one, the document is considered less relevant than the more recent ones. Lin and Hovy (2002b) introduced another penalty metric: the stigma words penalty. A predefined score value is removed from sentences starting with conjunctions, question marks, pronouns such as "he"/"she"/"they", and the verb "say" and its derivatives.

Once the sentences in the text(s) have been scored, the **generation** phase aims to order them and to select the ones that will compose the final summary, seeking to fulfill the compression rate. Also, a revision phase can be performed in order to improve the final summary readability.

Typically, the sentences are ordered considering their score. The ones with the highest score are selected until the compression rate is attained. Another common approach, specially in single-document summarization, is to consider each sentence position in the original text. However, ordering a summary whose sentences where retrieved from different sources is even more challenging. Sentences that are not related in any way, need to be ordered to form a readable text, so it is necessary to consider not only sentence position but also inter-document order at the same time. Considering this, the score must not be the best metric to define the summary order. Thus, other ordering strategies were proposed. Barzilay et al. (2002) experimented two ordering algorithms. While majority ordering takes into account the most frequent positions occurring in the original documents, chronological ordering orders the sentences by the publication date. Adopting a different approach, Lapata (2003) trained a probabilistic model using a corpus of domain specific texts. The model learns which sequences of features are likely to co-occur and makes predictions concerning preferred orderings. It learns constraints that exploit precedence relationships of verbs in the corpus, local coherence by tracking nouns to find relations between entities, and, finally, dependency relations that define the structure of the sentences. Okazaki et al. (2004), in turn, propose a method to refine sentence order

based on antecedent sentences. Prior to the application of this method, sentences are ordered in two steps: topical clustering – groups sentences on the same topic –, followed by chronological order – sorts sentences based on the source text publication date. The refining method is, then, applied over this initial ordering, in order to produce a well-structured text. This method considers precedence relations as a shortest path problem. *Distance*, a dissimilarity value of precedent information of a sentence. For instance, when a sentence has antecedent sentences and their content is not mentioned by previously arranged sentences, this distance will be high. This metric is used to decide whether a sentence should replace another sentence in the final order or not. Finally, Bollegala et al. (2006) present a bottom-up approach composed by four criteria that aim to capture associations between sentences. These criteria are combined into one criterion using a supervised learning approach. In addition to the criteria used in previous mentioned works ((Barzilay et al., 2002) and (Okazaki et al., 2004)), such as chronological order, topical closeness, and precedence, Bollegala et al. (2006) also use succession, a metric that assesses the coverage of the information if a given sentence is arranged after another. The strongest the association is, the closest the sentences must be in the summary.

After ordering has been done, in the majority of summarization systems, no further processing is done. However, the extracting nature of the process introduces the need of correcting cohesion, coherence, and fluency flaws, in order to produce a more readable text. For instance, these flaws include the lack of conjunctions between sentences (e.g. "however"), adverbial particles (e.g. "too"), and lack of information due to dangling anaphors (e.g. "in such a situation"), or proper names without adequate descriptions. Also, syntactically complex sentences can be hard to understand and can include extraneous information, contributing to a complex summary.

Several methods can be applied to the final summary to improve its readability, for instance anaphora resolution or concept generalization. Anaphora resolution aims to find the referent of an expression. By extracting sentences from their context, references can be missed. Identifying these referents can improve summary quality. Once identified the anaphoric sentence, a sentence that starts with a pronoun, for instance, a simple approach can be the inclusion in the summary of the previous sentence. Other methods include more sophisticated algorithms that exploit syntax and semantic knowledge. Concept generalization seeks to simplify context-specific concepts into more general ones. This process is usually performed using ontologies.

Extraction based methods assign scores to the document content units to determine relevant information to compose a summary. Those units are then retrieved from the original text(s) and combined to form the summary. Finally, a revision of the final summary can be performed to improve its quality.

2.1.3 Corpus-based methods

As soon as annotated corpora have been made available, researchers shifted their attention to apply machine learning techniques to annotated corpora, as a way of creating extracts. Elements (usually sentences) from each source document are extracted, analyzed in terms of features of interest and then labeled by comparison with the corresponding summary. The labeled feature vectors are fed to a learning algorithm. This algorithm emerges with rules that can be used to classify each sentence of a test document determining whether the sentence should be added to the summary or not. Therefore, the most significant differences among systems using this approach lie both on the selection of the learning algorithm and the selection of the features used to train it.

(Kupiec et al., 1995) was the first work using this approach. This work is based on a discrete feature set, which includes: sentence length cut-off – short sentences tend not to be included in summaries; fixed-phrase – sentences containing any of a list of fixed phrases (e.g., "this letter...", "In conclusion..." etc.), or occurring immediately after a section heading containing a keyword, such as "conclusions", "results", "summary" and "discussion", are more likely to be in summaries; paragraph – distinguishes sentences according to whether they are paragraph-initial, paragraph-final, and paragraph-medial; thematic words – the most frequent words are defined as thematic words; and uppercase words – proper names are often important.

Thereafter, Teufel and Moens (1999) investigated the use of the sentence rhetorical role in creating summaries of scientific articles. The features used as indicators of salience included: indicator phrases – meta-comments and argumentative phrases such as "we argue" or "in this article"; rhetorics indicator – indicates to which rhetorical class the indicator phrase belongs; relative location – sentences that occur in salient physical locations in the text; sentence length – rewards long sentences; thematic word – most frequent words are the most important ones; title – assigns higher weight to sentences with title words; and header type – looks at prototypical header keywords.

Conroy and O'Leary (2001) proposed the following features to train their classifier: the position of the sentence in the document, the number of terms in the sentence, and how likely sentence terms are given the document terms.

To capture the association and order of two textual segments (e.g. sentences), Bollegala et al. (2006) defined four criteria: chronology – arranges sentences considering the publication date of the document; topical-closeness – the association of two segments, based on the topical similarity; precedence – substitution of a segment that should be preceded by another segment; and succession – replacement of a segment that should follow another segment.

Concerning the learning algorithm, both Kupiec et al. (1995) and Teufel and Moens (1999) used a Bayesian classifier. This classifier combines the selected features to compute a probability for each sentence that determines whether it should be included in the summary or not. Conroy and O'Leary (2001) used a Hidden Markov Model (HMM), which given a set of features computes an a posteriori probability that each sentence is a summary sentence. Bollegala et al. (2006) used a Support Vector Machine (SVM) algorithm to analyze the data and yield the association strength between two segments, in order to determine if they should be included in the summary. Also, Fuentes et al. (2007) applied a SVM to a query-focused summarization system, using a set of structural, cohesion-based and query-dependent features.

Wong et al. (2008) combined supervised and unsupervised learning methods to produce extractive summaries. Sentence position, number of words in the sentence, high frequency terms are some examples of the features used in this work.

Finally, Hong and Nenkova (2014) proposed a supervised model for predicting word importance. By analyzing an annotated corpus of pairs containing abstracts and articles, they essayed to identify the words that tend to be preserved in the abstracts. A supervised framework was used to compute a set of indicators of importance. They concluded that features related to global importance, sentiment and topical categories helped to improve the estimation of word importance in a summarization context.

A *corpus*-based approach is a very adaptable approach, once features are known and defined. This approach, considering its nature, is easily tested. Also, it can make use of a wide variety of algorithms that are expected to produce different (and better) results. However, it has many weaknesses that are difficult to overcome. For instance, the creation of a corpus that aligns source documents with ideal summaries is a hard task, as the

ideal summaries have to be built by a human abstractor. Also, the domain dependency associated to the training corpus is a drawback that restricts the use of the system.

2.1.4 Abstraction methods

As language tools have evolved – new corpora, syntactic and dependency parsers, POS taggers have been created –, text summarization systems could make use of such resources to create better summaries, composed by rewritten sentences, generating abstracts. Thus, abstraction techniques could finally be used, and most research on summarization followed that path, in order to improve the final summaries.

Abstraction is typically performed in three steps. First, a semantic representation of the sentences in the input text is built. Then, selection, aggregation and generalization operations are executed on those semantic representations creating new ones. Finally, the new representations are rendered in natural language, by the generation of the sentences that will compose the summary.

Different approaches can be carried out when building abstract summaries. **Template filling** approaches are based on previously handcrafted templates whose slots determine the information that should be included in the summary. The slots in the template represent the salient information that is to be instantiated from the text. Afterwards, the template is rendered into natural language output. Radev and McKeown (1998) used a set of templates containing the salient facts reported in news articles. Once all the templates have been filled, relationships between templates are identified. Afterwards, a set of operations is performed over the templates. These operations can include for instance information merging or refinement. The resulting templates are then rendered into sentences. This approach provides high levels of compression, since each template, which can be filled using information from many sources, supports the generation of a single sentence. However, it is highly domain dependent considering that the templates are built taking into account the corpora to be summarized.

Other abstraction methods rely on **linguistic theories** to perform summarization. A popular theory used is Rhetorical Structure Theory (RST) (Mann and Thompson, 1998). RST was originally intended to describe texts, rather than the process of creating or reading and understanding them. It builds a tree structure of the input text and models rhetorical relations between textual units.

RST was primarily used to built a rhetorical parser (Marcu, 1997), that enables the derivation of rhetorical structure of unrestricted natural language texts. Afterwards, Marcu and Gerber (2001) explored its RST-based parser to a summarization system, in order to support the determination the most important units in a text. Each document is thus parsed and partitioned into a linear sequence of non-overlapping elementary discourse units (EDUs) – text fragments about a size of a clause. A discourse tree structure that subsumes the entire text is then built. Its leaves are EDUs which importance is measured by its proximity to the root. The EDUs closest to the root are considered more important. The next step consists in applying a clustering algorithm that groups similar EDUs, in order to remove redundant information. The initial clusters are ranked using a score function that is a combination of four values: (1) the average position of the EDUs; (2) the average importance of the EDUs; (3) the cluster size; and (4) the degree of similarity within the cluster. The highest score of a unit, the more important that unit is in the text. Finally, the summary is generated considering the most representative EDU from the most important cluster, and the subsequent EDUs from less important clusters until a target summary size is reached.

Another approach was developed by Pardo and Rino (2001), when creating an automatic summarization system for Portuguese. Their system is based on a theory that aims to model the discourse by combining semantic, rhetorical and intentional knowledge, in order to generate coherent summaries. Primarily focusing on text planning, this model takes into account three distinct representational levels, namely: intentional, semantic, and rhetorical. This system produces RST trees by mapping semantic and intentional relations onto rhetorical ones. These trees reflect the possible text plans. The linguistic realization of the summary is carried out by hand, resulting in the summary itself.

A different strategy uses *lexical chains* to detect the main topics of a document. Lexical chains are sequences of semantically related words that provide a model of the lexical cohesive structure of a text through a graph representation. By analyzing the graph's connectivity the strongest cohesive chains can be determined, and the sentences that represent these chains are selected to be part of the summary. A pioneer work on lexical chains was the one of Barzilay and Elhadad (1997). They used lexical chains as a technique to produce a summary without requiring its full semantic meaning. Instead, they rely on a model that represents the topic progression in the text derived from lexical chains. In order to find those lexical chains in the text, they use several knowledge sources, such as

a WordNet thesaurus, a part-of-speech tagger, a shallow parser and a segmentation algorithm. Strong lexical chains are then identified based on the connectivity between all the chains. Barzilay and Elhadad (1997) used several formal measures to identify strong chains, including: chain length, distribution in the text, text span covered by the chain, density, graph topology (diameter of the graph of the words) and number of repetitions. Considering those chains, significant sentences are the ones that contain the first appearance of a representative chain member in the text. These are the sentences selected to compose the summary. By using this same approach, Medelyan (2007) reported that lexical chains improve the performance of not only a single-document summarization, but also keyphrase indexing.

Also Ercan and Cicekli (2008) used lexical chains to improve an extraction based summarization system. They combined the formal measures used by previous works [Barzilay and Elhadad (1997) and (Medelyan, 2007)] with a clustering step. After identifying the strongest chains, they are clustered using co-occurrence metrics to determine the topics present in the text. A sentence from each topic is selected until the number of sentences of the summary has been reached. By using this strategy, they improve their results when compared to the previously reported approaches.

More recently, Fang and Teufel (2014) implemented a summarizer based on a model (Kintsch and van Dijk, 1978) that maps human comprehension to summarization. This model proposes the processing of propositions on a sentence-by-sentence basis, detecting argument overlap, and creating a summary on the basis of the best connected propositions. The summarizer that Fang and Teufel (2014) created analyzes the text looking for concepts by using co-reference resolution, named entity detection and semantic similarity detection. By using distributional semantics strategies and constructing a coherence tree, they were able to create a compositional semantic version of the input text in order to generate an understandable text from propositions. When evaluated using *ROUGE-L*, this summarizer achieved an f-measure of 0.418 for long texts, and 0.333 for short texts.

Abstraction approaches are expensive tasks, mainly because the output summaries are not as good as they were expected to be, specially considering the computational effort that has been done. Not only language tools were not sufficiently mature to deal with specific summarization challenges, but mainly the summaries produced did not improve the results of the systems in a way that abstracts could replace extracts. Thus, abstraction still remains a challenging task.

2.1.5 Evaluation

Like for any other piece of software, when developing an automatic summarization system, evaluation is of key importance. It provides a test to confirm or refute an hypothesis, or it can lead to additional hypotheses. All in all, evaluation provides a strategy for empirically testing a summarization approach at various stages of development (Mani, 2001a).

Evaluation aims to measure the system performance, to identify the contribution of a component to an overall system performance, and to adjust system parameters. Since summarization involves a machine producing an output in natural language, its evaluation is a complex problem to deal with, mainly because there is no obvious "ideal" or "correct" summary for a given input text or texts. Indeed, it is hard to define the notion of what is a correct output in terms of summaries. Yet, "ideal" summaries are used to evaluate automatic summarization systems, despite the possibility that the system generates a good summary that may be quite different from any human summary used as an approximation to the correct output. Since requiring humans to judge the systems output greatly increases the costs associated to evaluation, an evaluation which could use an automatic scoring program instead of human judgments is still very important, as this supports an easily repeatable process. But this requires the construction of standard sets of data involving human judgment, frequently named ideal (or gold) summaries, which consist of an example of correct summaries for the given text (or set of texts).

Be that as it may, manual evaluation is also used and is performed using surveys that must be answered by users after they have experimented the system, to determine whether the system output meets their needs or not.

Furthermore, as summarization involves compression, it is important to be able to evaluate summaries at different compression rates. This is a frequently neglected issue since it increases the scale and the complexity of the evaluation.

Evaluation methods and metrics currently in use for automatic summarization will be discussed in the following sections.

Methods

Despite all issues that make it difficult to evaluate an automatic summarization system, several metrics have been used. An automatic evaluation of a summarization system consists in two distinct methods: intrinsic and extrinsic.

Intrinsic methods test the system in itself with respect its inner functionality. These methods seek to assess the coherence, the cohesion, the informativeness, and the text organization of the generated summaries. They include both objective and subjective metrics.

Objective metrics measure time or accuracy in performing specific tasks. A summarization system can also be evaluated concerning the content of the summary. This content is compared sentence by sentence or fragment by fragment to one or more human-made ideal summaries.

Precision, recall and f-measure are objective metrics that seek to evaluate a summary in comparison to its corresponding ideal summary. These metrics are commonly used in the evaluation of natural language processing systems.

Precision represents the percentage of correct sentences in the automatic summary. These sentences are considered correct since they are present in the ideal summary. It is given by Equation 2.1:

$$P = \frac{number\ of\ sentences\ from\ the\ automatic\ summary\ that\ are\ also\ in\ the\ ideal\ summary}{number\ of\ sentences\ in\ the\ automatic\ summary} \tag{2.1}$$

Recall is defined as the percentage of relevant sentences in the automatic summary, considering the ideal summary sentences. It is given by Equation 2.2.

$$R = \frac{number\ of\ sentences\ from\ the\ automatic\ summary\ that\ are\ also\ in\ the\ ideal\ summary}{number\ of\ sentences\ in\ the\ ideal\ summary} \tag{2.2}$$

F-measure combines precision and recall by computing their weighted harmonic mean and aims at measuring the efficiency of the system. It is given by Equation 2.3.

$$f\text{-}measure = \frac{2 \times P \times R}{P + R} \tag{2.3}$$

Subjective metrics, in turn, judge fidelity to source, topic preservation, informativeness, readability, cohesion, coherence, etc. These subjective metrics aim at determining the quality of the summary. The evaluation of these features is commonly performed by using surveys, where users are asked to rate the automatic summaries, taking into

account those subjective metrics. Despite several studies have asked human users to answer those kind of surveys [(Brandow et al., 1995), (Minel et al., 1997), (Teufel, 2001), (Mani et al., 2002), (Kolluru and Gotoh, 2005)], there is no standard or commonly agreed survey method or task that defines specifically how this sort of evaluation is done. Brandow et al. (1995), for instance, did a large scale experiment where they tested, through a survey, the acceptance rate of single-document summaries, based on readability and content adequacy, with respect to the original text. Minel et al. (1997) used argument structure questions to infer the quality of the generated summaries. Teufel (2001) evaluated scientific papers by asking questions about the relatedness of current papers to prior research papers about the same topic. Mani et al. (2002) asked questions about significant source content that should be answerable from summaries.

In fact, as Kolluru and Gotoh (2005) point out by highlighting human subjectivity underlying the authoring of summaries, the inherent subjectivity behind this task substantiates its complexity, not only in terms of the definition of the task, but also in terms of its execution and evaluation.

On the other hand, **extrinsic methods** test the contribution of the system in relation to some other task, e.g. question answering, document retrieval or text classification, as summarization can be a module of that system. Thus, the main system is evaluated both with and without the summarization module to determine to what extent the later may improve the main system where it is embedded.

Evaluation plays a major role in the advancement of automatic summarization systems, although it requires more research in itself, particularly in developing cost-effective and user-centered methods. This task is still a major weakness in the development of an automatic summarization system, since manual evaluation is most of the times unfeasible (above all during development time) and the metrics developed to perform it automatically are useful but far from being satisfactory or consensual.

Automatic metrics

Many studies proposed metrics to support the automation of the evaluation task, allowing the systems to be tested during development time and improved on the basis of the results achieved. A number of evaluation techniques have been proposed to automatically judge the informativeness of the generated summaries.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a package of metrics, proposed by Lin (2004), which seeks to determine the quality of a summary by comparison with ideal summaries, previously established by human experts. It is composed by a set of metrics based on the overlapping units between the texts. The ROUGE package contains five evaluation metrics.

ROUGE-N (N-gram Co-Occurrence Statistics) is an n-gram recall between a candidate summary and a set of ideal summaries, and is computed using Equation 2.4.

$$ROUGE\text{-}N = \frac{\sum_{S \in IdealSummaries} \sum_{gram_n} Count_{match}(gram_n)}{\sum_{S \in IdealSummaries} \sum_{gram_n} Count(gram_n)} \tag{2.4}$$

$Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and in a set of ideal summaries, while n is the length of the n-gram. It is important to note that when adding more ideal summaries, the number in the denominator increases. This is reasonable since there might exist multiple good summaries.

ROUGE-L (Longest Common Subsequence) identifies the common subsequences between two sequences. Given two sequences X and Y, the longest common subsequence (LCS) of X and Y is a common subsequence with maximum length. The longer the LCS of two summary sentences is, the more similar the two summaries are. This metric has two major advantages: (1) it does not require consecutive matches but matches in-sequence that reflect sentence level word order; (2) as the longest in-sequence common n-gram is included, no predefined n-gram length is necessary. When applying LCS to summary-level, the union of LCS matches between a summary sentence and every candidate summary sentence is considered.

ROUGE-W (Weighted Longest Common Subsequence) favors sequences with consecutive matches. The basic LCS does not differentiate $LCSes$ that occur in-sequence from those that are not in-sequence. For instance, consider the following sequences:

X: [A B C D E F G]

Y_1: [A B C D H I K]

Y_2: [A H B K C I D]

When compared with X, Y_1 and Y_2 have the same *ROUGE-L* score. However, in what concerns *ROUGE-W*, due to its consecutive matches, Y_1 should be considered a better sequence to be chosen. In order to favor the in-sequence $LCSes$, the length of consecutive matches is stored, and each LCS is weighted considering that length.

ROUGE-S (Skip-Bigram Co-Occurrence Statistics) counts any pair of words in their sentence order, allowing for arbitrary gaps. It measures the overlap of skip-bigrams between a candidate sentence and a set of gold summaries sentences. Unlike LCS which only counts one longest common subsequence, skip-bigram counts all in-order matching word pairs. However, *ROUGE-S* does not give any credit to a candidate sentence if the sentence does not have any word pair co-occurring with its references. Thus, an extension to this metric was needed, the *ROUGE-SU*, which counts not only any pair of words in their sentence order, but also unigram occurrences of the words. This metric aims to differentiate candidate sentences that have words in common with the ideal summaries sentences, from sentences that do not have a single word co-occurrence with the ideal summaries sentences.

Document Understanding Conferences (DUC[1]) are evaluation workshops aiming at evaluating automatic summarization systems. These conferences provide datasets containing source texts, human summaries for those source texts and human judgments for the summaries built by the summarization systems in evaluation. Over et al. (2007) evaluated the effectiveness of ROUGE metrics has been evaluated by computing the correlation between ROUGE assigned summary scores and human assigned summary scores from DUC datasets of query-focused multi-document summaries. Considering these summaries, judges were asked to rate the *responsiveness* of the summaries, indicating how well a summary satisfies a given information need. Afterwards, average correlations of 78% (Spearman) and 84% (Pearson) were obtained between the human grades and the average values of all summaries for both *ROUGE-2* and *ROUGE-SU* (for 4-grams). These correlations suggest that ROUGE can be used effectively while developing automatic summarization systems, though when trying to detect real differences in system performance, the automatic methods may have less discriminative power, suggesting a human evaluation procedure to cope with this.

On a par with ROUGE, the **PYRAMID METHOD**, proposed by Nenkova and Passonneau (2005), is a method for evaluating content selection in summarization. It quantifies the relative importance of facts to be conveyed, involving the matching of semantic units. This method is oriented to evaluate multi-document summarization systems since it assesses that there is no single best ideal summary for a collection of documents, but rather that there is a set of ideal summaries to be confronted with an automatically generated

[1] See `http://duc.nist.gov/`

summary. This approach aims to perform a meaningful content evaluation with more informative scores, that can be a useful contribution to find which important information is missing in the summary.

The PYRAMID METHOD is based on the concept of summary content units (SCUs). SCUs are semantically motivated sub-sentential units. They are variable in length but not bigger than a sentential clause. An SCU consists of a set of contributors that, in their sentential contexts, express the same semantic context. SCUs denote information that is repeated across the ideal summaries.

Summaries are manually annotated with SCUs. Each SCU has a unique index, a weight, and a natural language label. The annotation starts with identifying similar sentences, and proceeds with finer grained inspection to identify more tightly related subparts. Contextual information from the entire summary is used to decide semantic equivalence of candidate SCUs. After the annotation process is completed, the final SCUs can be partitioned in a pyramid, that represents the opinions of multiple ideal summary writers each of whom have written an ideal summary for the input set of documents. The partition is based on the weight of the SCUs. Each pyramid tier contains all and only the SCUs with the same weight. For instance, when using annotations from four ideal summaries, the pyramid will contain four tiers. SCUs of weight 4 are placed in the top tier and the ones of weight 1 on the bottom. The highest the tier, the more occurrences of that SCU in the ideal summaries. That is, SCUs in tier 4 appear in the four ideal summaries considered. In short, the goal is to maximize information content value. So, an SCU from a lower tier should not be expressed if all the SCUs in the previous tier have not been expressed.

Based on a content pyramid, the informativeness of an automatic summary is computed as the ratio between the sum of the weights of its SCUs and the sum of the weights of an optimal summary with the same number of top scoring SCUs. This ratio defines the pyramid score, which quantifies the relative importance of the facts conveyed.

The PYRAMID method was evaluated using the dataset of DUC 2005, achieving a correlation of approximately 90% between the automatic and the human summary scores.

Finally, Giannakopoulos et al. (2008) defined **AUTOSUMMENG** , a method for the automatic evaluation of summarization systems. This method builds a graph representation for each summary, the automatic and the ideal one. Each graph is composed by character or word n-grams that are related through their spatial proximity within the text. The edges of the graph are labeled with the distance between two neighboring n-grams in the

original text. In order to evaluate two summaries, an automatic and an ideal one, a degree of similarity between them is obtained based on co-occurring edges in both graphs. At this point, generated summaries that have a higher degree of similarity with the ideal summaries are considered better. This method was tested in the corpus of DUC from 2005, 2006 and 2007. It achieved an average correlation of 89% between the automatic and the human scores for the three corpus in test.

2.1.6 Summarization in English

This section provides an overview of a set of extractive automatic text summarization systems for English, the language for which, by far, more studies on automatic summarization have been published in the literature. All these systems have inspired our research work, namely concerning the structure of our system and the metrics used. Along with the description of the methods carried out by these systems, an evaluation score, when available, is also reported.

SUMMARIST

Hovy and Lin (1999) developed SUMMARIST whose goal is to provide both extracts and abstracts for arbitrary English input text. It is based on the three typical steps of summarization: topic identification, interpretation and generation. This approach combines symbolic world knowledge – embodied in WordNet, dictionaries, and similar resources – with natural language processing – using information retrieval and statistical techniques.

In order to **identify the document topic**, several heuristics are used to filter the input to retain only the most important, central topics. These heuristics include the Optimal Position Policy (*OPP*) (described above in more detail in Section 2.1.2) that takes into account the position of the sentence in the original text, word frequencies and the presence or absence of cue-phrases in the sentence. They are then combined to produce the overall ranking for all sentences. The **topic interpretation** fuses topics into one (or more) unifying concept(s). In order to create those fused concepts, two methods are employed: the concept wavefront and the concept signature. Concept wavefront aims to count concepts instead of words by using a concept taxonomy, such as WordNet, for inter-concept relatedness links. The most appropriate concepts are found in the taxonomy through a wavefront of interest. This wavefront is defined by a set of nodes representing concepts.

These concepts can all be generalized by a broader concept (e.g. the words "vegetables", "fruit", "bread", and "milk" can be generalized to the concept "groceries"). A concept signature (described in Section 2.1.2 in more detail), in turn, is a topic word (the head) together with a list of associated pairs (word, score), where each score (computed using *tf-idf*) provides the relative strength of association of its word to the head.

The final step in the summarization process is to **generate the summary**, consisting of those fused concepts. SUMMARIST contains three generation modules, associated with different levels of application. Topic output generates a simple list of topics, when no text summary is needed. Phrase concatenation includes a rudimentary generator that composes noun phrase- and clause-sized units into sentences, to built the text. Finally, an external sentence planner, which takes as input the fused concepts, is used to generate an abstract summary.

Further work (Lin and Hovy, 2000) was done to improve this system. The method of topic signatures proved to be useful in the topic identification stage, since many texts may be about a given concept without ever mentioning this concept itself.

NEATS

Lin and Hovy (2002b) addressed multi-document summarization by developping NEATS. It is an extraction-based multi-document summarization system that, from a set of documents, seeks to extract relevant or interesting portions about some topic and present them in coherent order. It leverages techniques proved effective in single document summarization such as: term frequency, sentence position, stigma words, and maximum marginal relevancy (*MMR*) to select and filter content. Given an input of a collection of newspaper articles, NEATS generates summaries in three stages: content selection, filtering, and presentation.

The **content selection** stage identifies important concepts mentioned in the collection of documents. Key concepts are identified in unigrams, bigrams and trigrams using the likelihood ratio (Dunning, 1993). These concepts are then clustered in order to identify subtopics within the main topic. Then, each sentence in the document set is ranked, using an algorithm that rewards most specific concepts. Once the content selection stage is finished a list of ranked sentences is available.

Thereafter, the **content filtering** stage, which aims to reduce the number of candidate

sentences to enter the summary, applies three different filters: sentence position, stigma words, and *MMR*. The sentence position filter defines that the ten lead sentences are retained. Stigma words filter ignores sentences starting with conjunctions, question marks, pronouns such as "he"/"they", and the verb "say" and its derivatives. Finally, *MMR* (described in Section 2.1.2) considers the overlap between top sentences in order to reduce eventual redundancy.

At this stage cohesion and coherence problems tend to arise. In order to solve them, at the **content presentation** stage, for each sentence to be included in the summary, the system first locates and includes a suitable introductory sentence for each remaining sentence, in order to provide introduction and context information about what is coming next. Secondly, the pairs are reordered considering the chronological order and all the time expressions are disambiguated with explicit dates.

NEATS was evaluated in DUC 2001 (Lin and Hovy, 2002a), and achieved an f-measure of 0.45, while the baseline – which retrieved only the first sentences of each document – achieved an f-measure of 0.39, and the human summary attained an f-measure of 0.58.

NEWSBLASTER

NEWSBLASTER[2] (McKeown et al., 2001) is a summarization system of on-line news. The main focus of this system is to identify similarities and important differences across the input set of documents. This is a system composed by several modules each of which perform different strategies depending on the type of the documents in the input set.

NEWSBLASTER encompasses three main modules: a preprocessor, which transforms the incoming data into a specific representation; a router, that determines the type of each input document set, and directs the input texts to the summarizer; and the summarizer, which finally produces the output.

The **preprocessor** harmonizes the data that is originally in several different formats into a uniform format.

The **router** determines the type of the document set, among four possibilities: Single-event, Person-centered, Multi-event, Other – types which have been manually isolated with the help of a training corpus.

[2] `http://newsblaster.cs.columbia.edu/`

The **summarizer** consists of two systems: MULTIGEN (Barzilay et al., 1999) and DEMS (Schiffman et al., 2001) (Dissimilarity Engine for Multi-document Summarization).

MULTIGEN summarizes a specific type of input, which includes news articles presenting different descriptions of the same event. The system consists of a pipeline architecture that includes an analysis component – which integrates machine learning and statistical techniques to identify similar paragraphs, and similar phrases within paragraphs – and a generation component – to reformulate the summary wording. The analysis component breaks documents into smaller text units and computes a similarity metric across text units, regardless of the source document. Once similar paragraphs are identified, the generation component selects information to be organized as a coherent text.

As for DEMS, it was developed to handle the document sets that do not conform to single event descriptions. It manages two different types of documents: very loosely related documents, with a common topic, and biographies. Concerning very loosely related documents, it uses three main techniques to compute significance: identifying importance-signaling words through an analysis of lead sentences in a large corpus of news, identifying high-content verbs through a separate analysis of subject-verb pairs in a news corpus, and finding the dominant concepts in the input clusters of articles – rather than frequent words. With respect to biographies, the approach followed is data-driven, relying on discovering how people are actually described in news reports. Corpus statistics from a background corpus was used, along with linguistic knowledge to select and merge descriptions from a collection of documents.

Using a corpus of news articles specifically built for the sake of the evaluation (Mckeown et al., 2005), NEWSBLASTER achieved a pyramid score of 0.4377, while the human summaries obtained a pyramid score of 0.4390.

MEAD

MEAD (Radev et al., 2000) is a multi-document system that summarizes clusters of news articles automatically grouped by a topic detection system. It uses information in the clusters to select the sentences that are most likely to be relevant to the topic of the cluster.

The system receives a set of documents as input and through a Topic Detection and Tracking (TDT) system, called CIDR, clusters all the documents. CIDR uses modified *tf-idf* to produce clusters of news articles on the same event. An event cluster consists of

chronologically ordered news articles from multiple sources that describe an event as it develops over time. In order to apply CIDR to summarization, a technique called centroid-based summarization (CBS) was developed. This technique uses as input the centroids of the clusters produced by CIDR to identify which sentences are central to the topic of the respective cluster.

Afterwards, the cluster sentences are scored, on the basis of three features: centroid value, position value, and first-sentence overlap. The centroid value is a measure of centrality, and is computed as the sum of the centroid values (*tf-idf* scores) of all the words in the sentence. The positional value refers to the sentence position in the document. This value decreases linearly as the sentence gets more distant from the beginning of the document. The overlap value defines the sentence overlap over the first sentence of the document. It is computed as the inner product of the *tf-idf* weighted vectors of both sentences. The sentence score is then a linear combination of these three features subtracted with a redundancy penalty. This penalty is applied to each sentence which overlaps with sentences that have higher score values. Finally, the summary is generated containing the higher scored sentences, until a desired compression rate is reached.

CATS

CATS (Cats is an Answering Text Summarizer) (Farzindar et al., 2005) is a system for summarizing multiple documents concerning a given topic at a level of granularity specified in a user profile. It was developed to participate at DUC2005 and was built to meet the competition specific requirements.

The system receives as input a collection of news documents to be summarized and a set of questions, which define the topic to be addressed in the summaries. CATS includes five stages: question analysis, document analysis, sentence scoring, post-processing, and sentence selection.

Question analysis examines the question along two dimensions: named entities and basic elements. First, the semantic types of named entities is identified. The more a semantic type appears in the question the highest is the probability that this type of named entities should appear in the summary. Then, the question is split in basic elements, which consists of a triple describing the grammatical relationship between two words in a sentence. This decomposition in basic elements facilitates the sentence comparison.

Document analysis determines which information is important to include in the summary. This analysis addresses five points: temporal expressions, thematic segmentation, sentence segmentation, basic elements and named entities. First, the texts are split into paragraphs and, since documents are articles from newspapers, their publishing date is identified using a temporal information module. Based on each document publication date, relative temporal expressions are firstly resolved. Thematic segmentation determines which sentences pertain to each topic by *TextTiling* (Hearst, 1997), an external segmenter that works at paragraph level. Sentence segmentation divides paragraphs into sentences. Then, each sentence is split into basic elements and its named entities are detected and categorized by using the same techniques as used for the question.

The **sentence scoring** step includes two phases. The first phase filters sentences at thematic level, using a cosine similarity, computed over the question, to identify interesting segments. Then, sentences retrieved from the first phase are scored using a linear combination of seven measures, which include: (1) basic elements – comparison of the basic elements in the sentences with those of the question; (2) cosine similarity – measured between the sentences and the question; (3) weight of the sentence – sum of the weights (*tf-idf*) of its words; (4) absolute position in the text; (5) relative position in the text; (6) named entities – number of named entities in the sentence which have the same category as named entities in the question; (7) prototypical expressions – number of prototypical expressions in the sentence, which indicates that the sentence has a higher probability of containing important information. Finally, sentences are sorted in decreasing order of score. Also redundancy is addressed by computing cosine similarity between sentences, and discarding the ones that have higher similarity or that contain two or more common named entities.

In order to obtain a more concise and coherent summary, **post-processing** operations are performed to eliminate less important parts or replace expressions by more concise ones. So, temporal expressions resolution is the first operation performed, followed by a sentence compression task, which removes parenthetical expressions.

Finally, the **sentence selection** step elects sentences with the highest scores, until the summary contains at most 250 words. The sentences are then sorted by date in increasing order.

CATS was evaluated in DUC 2005 and achieved a recall value of 0.13 – defined by the mean of 50 summaries recall values – while the best system attained also 0.13.

SUMMA

SUMMA (Saggion, 2014) is a framework for developing text summarization applications. It has been implemented using the GATE API[3] so it relies on GATE documents, GATE annotations, and other data-structures for information storage and retrieval.

SUMMA contains several components for creating summarization applications. **NEs statistics** computes term frequency and *tf-idf* statistics for the input documents. **Vector computation** creates feature maps that associate terms (e.g. words) with their weights, in sentences, paragraphs, titles, sections or documents. **Title sentence similarity** computes the cosine similarity between any sentence in the document and any other textual unit in the same document (e.g. the title). **Position scorer** assigns a relevance score to the sentences based on their positions in the document. **Cue phrase scorer** defines a list of cue-phrases that contains words considered important and their weights, which are used to define the sentence score. **Query method scorer** computes the sentence scores based on the similarity of each sentence to an input query. **Term frequency scorer** defines a scoring function based on weights associated to terms in the sentences. **Semantic scorer** takes into account the named entities and types in the sentence, in order to set its relevance score. **N-Gram computation** implements customized n-grams, in order to support content-based evaluation using ROUGE. **Centroid** performs a centroid computation procedure for a corpus. The centroid is the average of all document vectors. **Sentence centroid similarity** uses cosine similarity to compute the similarity of each sentence in the document and the centroid of a set of documents. **Simple summarizer** creates a sentence scoring rank for a single document and selects the top ranked sentences according to a given compression. **Simple multi-document summarizer** performs summarization given a set of related documents which sentences have already been scored. It selects the top ranked sentences, applies a redundancy filter, groups the sentences from the same documents together and retrieves the sentences in the same order as they appear in the input documents.

2.1.7 Summarization in Portuguese

This section describes the research work on summarization for the Portuguese language. Different strategies have been carried out to perform not only extraction based sum-

[3] http://gate.ac.uk

maries but mainly abstraction ones. The first two systems described below (GISTSUMM and SUPOR-2) do not require deep linguistic knowledge to perform single-document summarization. While GISTSUMM relies in shallow statistical features to create extracts, SUPOR-2 relies in machine learning techniques to drive the content selection. The third one (CN-SUMM) also performs single-document summarization, though creating a deep representation of the input text from which the content of the summary is selected. Finally, the last one (CSTSUMM) relies on a theory for multi-document discourse structure. The systems and their methods are described below.

GISTSUMM

GISTSUMM (Pardo et al., 2003) is a single-document summarizer which aims at identifying in a text its main idea. It is based on the notion of gist, which is the most important passage of the text, conveyed by just one sentence, the one that best expresses the text main topic. The algorithm of the system relies on this sentence to produce extracts.

GISTSUMM comprises three processes, namely, segmentation, sentence ranking, and extract production. **Segmentation** delimits sentences, addressing them as minimal units. **Sentence ranking** scores sentences applying either the keywords method, or *tf-isf* (Term Frequency – Inverse Sentence Frequency), named average-keywords method, depending on the user's choice. Finally, in order to **create the summary**, the system averages the sentence scores, determining their threshold. The most important sentence of the text, the gist sentence, is the first one being added to the summary. Then, until the compression rate is achieved, sentences which (1) contain at least one word whose stem also corresponds to some word in the gist sentence, and (2) have scores above the threshold, are selected to form the summary. The compression rate is defined by the user and determines the summary length.

GISTSUMM was evaluated (Balage Filho and Pardo, 2007) using a corpus of 20 short scientific papers on Computer Science (Computation and Language corpus – *cmp-lg*), written in Brazilian Portuguese, made available by ACL (Association for Computational Linguistics). The summaries of the papers were used as the human summaries required by ROUGE metric. Building the summaries through the keywords method, GISTSUMM achieved an f-measure of 0.24 (computed with *ROUGE-1*). Using the average-keywords method, GISTSUMM attained a f-measure of 0.21 (computed with *ROUGE-1*).

SUPOR and SUPOR-2

SUPOR (Módolo, 2003) is an environment for automatic summarization of texts in Portuguese that once customized can produce several types of summaries depending on the summarization strategies selected. It combines several types of features that have been tested in English texts in order to find out which ones can best contribute to improve text summarization of Portuguese texts. This environment uses language-specific resources, namely electronic dictionaries, a lexicon and a thesaurus, a parser, a part-of-speech tagger, a steamer, and a sentence chunker. Different summarization strategies can be carried out by selecting different groups of features. Each strategy can be used as a step of a major analysis that aims to identify which combination of features improves the extraction of the text units that will compose the final summary. In addition, the framework includes a customizing step that allows a user to choose distinct summarization features and also different compression rates.

The very first goal of each strategy is to identify the relevant text units that will compose the extract summary. Four extractive approaches can be carried out using SUPOR.

The **Classifier** method uses a Bayesian classifier that aims to recognize relevant features in a text, by taking into account (1) the sentence length – limited to five words as the minimum; (2) the word frequency; (3) signaling nouns; (4) sentence or paragraph location; and (5) occurrence of proper nouns. The result of the training phase is a probabilistic distribution that helps the summarizer to select sentences that meet these features.

The **Lexical Chains** method computes lexical chains based on correlations of words. Nouns are the basic units of a source text. In order to compute the lexical chains, SUPOR uses an ontology and WordNet to identify lexical chaining mechanisms of cohesion (synonymy/antonym, hyperonymy/hyponymy, etc.). These mechanisms result in a set of strongly correlated words. Afterwards, three heuristics can be applied in order to select the sentences of the final summary. The first heuristic selects every sentence in the source text based on each member of every strong lexical chain. The second heuristic applies the former one only to representative members of a strong lexical chain – the members whose frequency is higher than the average frequency of all words in the chain. Finally, the third heuristic considers the representativeness of a given strong chain in the source text.

The **Relationship Map** method seeks to interconnect paragraphs to build maps of correlated text units. From these maps different paths can be produced. Dense paths, those

with more connections in the map, are built by choosing top-ranked paragraphs totally independent from each other. Deep paths are focused on semantically inter-related paragraphs, in order to improve the cohesion and coherence of the text. Finally, as both the previous mentioned methods are focused on a single topic, the segmented bushy paths are used to combine distinct topics.

The last method is the **Importance of Topics**, which uses *tf-isf* measure to identify sentences that convey the relevant topics to include in the summary. Text Tiling (Hearst, 1997) is the algorithm used to delimit the topics. In an initial assessment of its performance (Rino and Módolo, 2004), SUPOR was the first ranked system when compared to other available summarizers for Brazilian Portuguese, achieving an f-measure of 0.42.

Afterwards, another version of this system was created. SUPOR-2 (Leite and Rino, 2008) is a single-document extractive summarizer that combines linguistic and non-linguistic properties for extraction. It combines three methods defined in the SUPOR framework – lexical chains, relationship maps and importance of topics – with four features – sentence length, sentence location, occurrence of proper nouns, and word frequency. This version, SUPOR-2, was evaluated (Leite et al., 2007) against a modified version of TEXTRANK (Mihalcea and Tarau, 2004), a graph-based ranking model for text processing, using the corpus *TeMário* (Pardo and Rino, 2004). SUPOR-2 achieved a recall of 0.58, while TEXTRANK obtained a recall of 0.56.

CN-SUMM

CN-SUMM (Complex Networks-based Summarization) (Antiqueira, 2007) is a single-document summarizer that selects sentences for an extractive summary using complex networks. This method uses a network of sentences that requires surface text preprocessing – tokenization, post-tagging and lemmatization – thus allowing to assess extracts obtained with no sophisticated linguistic knowledge. The graph or network that represents each text is composed by nodes, corresponding to sentences, while edges connect sentences that share common meaningful nouns. After the network has been created, a subset of sentences (nodes) are selected to compose an extract by ranking them according to some network measurement. Network measurements include, for instance, the degree of a node – number of edges attached to the node; a clustering coefficient – number of neighboring edges connections; minimum paths; matching index – compares the connectivity

between two nodes linked by an edge; etc. Depending on the approach used to find the best path between the nodes, several strategies can be applied to the network in order to select the sentences that compose the final summary. Degree, shortest path, local index, d-rings, k-cores, w-cuts, communities and voting are the strategies that were developed.

In a later work, Antiqueira et al. (2009) introduced, for example, seven different network measurements that lead to different ranking mechanisms, that can be defined as different summarization strategies. All these strategies were evaluated and the best one achieved a f-measure of 0.43, when run over the *TeMário* corpus (Pardo and Rino, 2004).

CSTSUMM

CSTSUMM (Jorge and Pardo, 2012) is a CST-based multi-document summarizer. Cross-document Structure Theory (CST) (Radev, 2000) is a functional theory for multi-document discourse structure. It is used to describe the semantic connections among units of topically related documents. In their work, Jorge and Pardo (2012) focused on selecting content. They assume that a previous representation of the source texts according to the CST model has already been created, and they explore different content selection strategies upon this model.

First, the source texts are represented by a CST graph, manually built, where sentences are nodes and the edges of the graph are CST relations. Then, depending on the goal of the summary, an operator is run over the graph. An operator is defined by a set of rules, specifying conditions that, if verified, determine how the content of the summary is selected. These conditions are defined by CST relations that can be combined. Depending on the intent of the final summary, different operators are used. The "contextual information" operator, for instance, determines that the sentences selected to be in the summary must be connected by *historical background* or *elaboration* relations.

An operator receives an initial order of sentences. Their rules trigger actions that modify this order taking into account the CST relations considered most relevant for the operator. These actions create a new order of sentences, in which the compression rate is applied, defining the sentences that compose the final summary.

CSTSUMM was evaluated attaining an f-measure of 0.53, when run over the *CSTNews* corpus (Aleixo and Pardo, 2008).

2.1.8 Commercial applications

INTELLEXER SUMMARIZER[4] is a desktop application that analyzes the document, extracts its main topics and puts them into a short summary. It is able to create theme-oriented (e.g. politics, economics), structure-oriented (e.g. scientific article, patent) and concept-oriented summaries.

This solution is announced as being based on a semantic approach. It performs a semantic analysis of the text by extracting Verb-Object pairs and defining the relationship between them. Then, the semantic weight of each pair is determined and the summary is created accordingly. If the summary type is specified, the system will place primary importance upon specific concepts and structure, thus making the summary theme- or structure-oriented.

Intellexer Document Summarizer is developed by `EffectiveSoft`[5] and is available in the web for free trial and for purchase.

SUBJECT SEARCH SUMMARIZER™, or SSSUMMARIZER, creates brief summaries of virtually any document or Web page and translates them into a chosen language. It generates and displays summaries as a list of key sentences, extracted from documents using Kryloff's proprietary Subject Search technology. It presents and translates sentences which reflect the subject of a given document. SSSUMMARIZER is capable of processing textual information of any length, on any subject, in any one of about 40 languages. Furthermore, it can automatically detect the input document language.

SSSUMMARIZER has been developed by `Kryloff Technologies`[6] and is available in the web for free trial and for purchase.

SUMMLY[7] is a mobile application designed for *iOS*, available for free. It is intended to summarize popular news stories that the reader can easily digest.

It identifies key points that matter most and summarizes them into a summary of 400-characters, through a genetic algorithm that mimics the human thinking, using organic metrics to extract the most critical components of the news texts.

[4] `http://summarizer.intellexer.com/`
[5] `http://www.effectivesoft.com/`
[6] `http://www.kryltech.com/`
[7] `http://summly.com/index.html`

2.1.9 Summary

Text summarization is a very challenging task. Extraction based summarization still poses several problems related to the coherence and cohesion of the output text. Abstraction methods, on the other hand, are highly domain-dependent since they rely on corpora from specific domains to be trained. Also, due to its intrinsic complex nature, these methods still face many challenges, specially the creation of readable summaries that satisfy users needs.

Concerning multi-document summarization, the approaches to identify relevant content and to remove redundant one have been proved to be useful. However, the resulting summaries are likely to be incoherent, since revision strategies have not been carried out carefully.

2.2 Sentence reduction

Recall from Chapter 1 that our aim is to post-process an extractive summary. One of the steps of post-processing is *sentence reduction*, which is introduced in this Section, in the context of the broader research area of text simplification. Sentence reduction is, in fact, a specific form of text simplification. It can be defined as syntactic simplification applied to one sentence at a time. This research area is also referred in the literature as sentence compression or sentence simplification.

Firstly, some definitions will be presented and the most common approaches will be described. Afterwards, the methods currently used in sentence reduction will be detailed.

2.2.1 Definitions

Long and complex sentences tend not only to be difficult to process for human readers but they also may raise various problems in automatic language processing systems. Complicated sentences can, for instance, lead to confusion in technical texts, such as assembly manuals, user manuals or maintenance manuals for complex equipment (Chandrasekar and Srinivas, 1997).

Text simplification is a Natural Language Processing (NLP) task that aims at clarifying natural language texts by simplifying its sentences structurally into shorter and simpler sentences, and by preserving at the same time the meaning and the information con-

tained within them as much as possible. Accordingly, text simplification may ease automatic text processing tasks and comprehension of the text by humans.

Applications such as parsing, machine translation, information retrieval, and text summarization can benefit from text simplification. When input sentences become long and complex, the performance of these applications tends to worsen (Chandrasekar et al., 1996). Ambiguity in long sentences can also be an issue at stake. If the sentences are transformed into shorter and simpler ones before they are fed in to another module, the level of ambiguity may be reduced and the system performance can be expected to improve and be less error-prone (Feng, 2008). Many of the problems raised by complicated sentences are either eliminated or substantially reduced in its simplified form.

Text simplification can thus be not only a part of a natural language processing task but also a stand-alone application, whose target is human readers, specially people with some language disadvantage or disabilities, such as illiteracy or aphasia – difficulty in producing or comprehending spoken or written language –, hearing impairments, or other intellectual disabilities. Regular texts often require high level of linguistic skills and sufficient real world knowledge that these individuals often lack. Providing lexical simplification and/or reducing linguistic complexity of written texts would make the latter more comprehensible to these individuals. Moreover, a simplified text could help these individuals feel more connected to the community by reading and understanding what is going on in the world surrounding them (Feng, 2008).

Linguistic issues

Text simplification deals with many different linguistic issues depending on the type of simplification performed: lexical, syntactic or discourse simplification. Example 2.1 shows a sentence and its simplified version.

Example 2.1: Simplified sentence.

Original sentence:	**Simplified sentence:**
A casa, que os Maias vieram habitar em Lisboa, no Outono de 1875, era conhecida pela Casa do Ramalhete.	*A casa era conhecida pela Casa do Ramalhete.*
The house in Lisbon to which the Maias moved in the autumn of 1875, was known as Casa do Ramalhete.	The house was known as Casa do Ramalhete.

The **lexical simplification** task involves replacing "difficult" words with their simpler

synonyms. However, problems arise when faced with the identification of those "difficult" words and how these words should be replaced. The most common approach to identify difficult words is to use word frequency. In this context, word frequency is computed considering the frequency of the word in typical language usage. A word with lower frequency indicates that the word is not commonly used and thus it is likely a difficult word to be processed by humans. Another approach is to treat each content word as being equally difficult and let the user decide which words need to be simplified. The replacement of a difficult word can be done by paraphrasing, either through a lexical database (Fellbaum, 1998), or by using a predefined vocabulary list or dictionary definitions.

Syntactic simplification is a much more complex task than lexical simplification since the input texts must be analyzed to produce detailed tree-structure representations suitable for syntax transformation (Feng, 2008). This procedure is typically performed in two steps: first, simplification candidate structures are identified; then, a set of simplification rules is applied.

The identification of structures candidates to simplification is made by marking particular linguistic features, which include coordinated or subordinated conjunctions, relative pronouns, clause or phrase boundaries, or noun phrases to which clauses are attached. Several factors must be considered. Relationships between word groups contained in the sentence should be analyzed to provide basic information such as subject, verb and object in order to form the new simpler sentences. For instance, to convert a passive construct into its active form, the agent of the action needs to be identified and verbs need to be annotated with grammatical information such as voice, tense and aspect, for them to be change, if necessary, upon the application of the simplification procedure. Therefore, in order to provide the required information when simplifying the identified constructs, further analysis is needed. The syntactic analysis required for simplification is available through many NLP technologies, such as part-of-speech (POS) taggers, and constituency or dependency parsers.

The second phase of the syntactic simplification aims to define rules that determine the structures that should be simplified. These rules can either be hand-crafted or automatically generated.

Concerning **discourse simplification**, the major linguistic issue is to maintain coherence and cohesion of the simplified text. While syntactic simplification is applied to one sentence at a time, discourse simplification aims to consider the interactions across sen-

tences. Therefore, the most common approach is to perform discourse simplification after sentence syntactic simplification in order to correct possible cohesion errors. For instance, when changing a sentence from passive to active voice, the order in which the noun or pronoun is introduced into the discourse is changed. Due to this transformation, the correct link of pronouns and their referring expressions across sentences may be broken. Links between sentences should be corrected in order to build a coherent text. Consequently, preserving anaphoric cohesion, not only by performing anaphora resolution and replacement, but also by ordering simplified sentences, are the main tasks carried out to address the lack of cohesion of simplified texts.

2.2.2 Approaches

Text simplification is a recent research area in Natural Language Processing. Different systems have experimented different approaches, and there is no common approach of building such a system. As a consequence, there are no ground rules to be followed, but instead some techniques that were tested, which include rule induction from corpus training, or manual development of syntactic rules which define what to be simplified.

Moreover, the approach to be used also depends on the purpose of the system. Different approaches are used in case the system aims to optimize the performance of another natural language processing application or if it is intended to serve as reading assistance for human readers. Also, depending on the specific purpose of each system, linguistic issues are addressed at different dimensions and different levels of depth.

Be it as it may, most of the approaches are mainly based on syntactic level transformations. Thus, most previous research have not considered that much the discourse-level issues that arise from applying syntactic transformations at the sentence level.

Current text simplification systems fall in two categories: those that aim at being a useful tool to improve the performance of other automatic NLP applications, and those intended to serve as reading assistance for human readers.

Since this work aims to use a sentence reduction system as a complement of another NLP task (summarization), the focus here will be mainly on studies that addressed text simplification, or specific sentence reduction as a part of another system (mainly summarization systems).

Depending on the specific purposes of each system, different levels of linguistic is-

sues are addressed and different approaches are pursued. Approaches can be divided in three types: rule-based, formally represented and machine learning approaches. These approaches are described in the following sections. Evaluation results for each approach are reported when available.

Rule-based approaches

Previous works by Chandrasekar et al. (1996) and by Jing (2000) have focused on syntactic simplification. These works target specific types of structures identified using rules that have been induced through an aligned corpus of complex and simplified texts.

Chandrasekar et al. (1996) developed a syntactic simplification system to improve the performance of a full parser. The simplification is accomplished in two steps: analysis and transformation. The analysis stage provides a structural representation of the input text while the transformation stage uses this representation to identify simplifiable constructs and apply simplification rules to simplify them.

In particular, a set of articulation points where sentences can be split were defined. These points include punctuation marks, subordination and coordinating conjunctions, relative pronouns, and the beginning and ends of clauses and phrases. Segments of a sentence between two articulation points may be extracted as simplified sentences.

Based on the articulation points, and by mapping sentence patterns to simplified sentences, a set of transformation rules are defined. As the system aimed at processing texts from different domains, the simplification rules must be as general as a generic-domain system requires. Hence, a method to develop rules which can be easily induced for a new domain was proposed (Chandrasekar and Srinivas, 1997). These rules were induced using an annotated aligned corpus of complex and simple text. In addition, gap-filling routines were also developed in order to correct possible errors in the sentences due to simplification. Despite having raised many discourse issues, those were not addressed in their work. Finally, no evaluation has been reported when regarding this method.

Jing (2000) created a simplification system that decides in five steps if a sentence should be reduced. The first step builds a syntactic parse tree for the sentence, while annotates each node in the parse tree with context information. Afterwards, a grammar checking step marks the nodes that cannot be removed, as this removal would compromise the grammaticality of the sentence. In the third step, the system decides which com-

ponents in the sentence are most related to the main topic. The importance of a phrase in the local context is measured based on lexical links between words. Then, based on a corpus of sentences reduced by human abstractors, a probability of reducing a certain phrase is obtained by considering the phrases that were removed by humans. The final step decides, based on the results from previous steps, which phrases should be removed, reduced or kept unchanged. A phrase is removed only if it is not grammatically obligatory, it is not the focus of the local context and has a reasonable probability of being removed by humans. This simplification procedure was included in a single-document summarizer (Jing and McKeown, 2000). Once the key sentences in the input document have been identified, the sentence reduction module removes extraneous phrases from each sentence. The sentence reduction procedure was evaluated in a corpus of 500 sentences and their reduced forms in human-written abstracts. The results show that 81.3% of the reduction decisions made by the system agreed with those of the humans.

Closer to our work are the work of Siddharthan (2003) and Zajic et al. (2007).

Siddharthan (2003) has built a text simplification system which aimed at providing not only a complete formalism on syntactic simplification, but also to address discourse issues left open in previous research.

This system comprises three stages: analysis, transformation, and regeneration. The analysis phase aims at preparing the text to transformation. The text is segmented into sentences and then annotated. Clauses and phrases that can be simplified are identified, as well as pronouns and their co-referents.

The analysis phase output is then passed to the transformation phase, which applies syntactic simplification rules to the analyzed text. These hand-crafted rules that specify relations between simplified sentences are applied sequentially within each sentence.

Finally, the regeneration phase includes two modules: one that handles conjunctive cohesion, and other that handles anaphoric cohesion. The conjunctive cohesion module orders the simplified sentences, inserts cue-words between them, and resolves determiners – since simplification may result in the duplication of noun phrases. The anaphoric cohesion module aims to correct broken pronominal links, by replacing pronouns with a referring expression for its antecedent noun phrase. The correctness of text simplification has been assessed by human judges that considered that 67% of the simplified texts are grammatically correct and preserve the meaning of the original text.

This simplification procedure was then used to improve content selection in a sum-

marization context (Siddharthan et al., 2004), that is before extracting sentences to be summarized during a clustering phase. First, parenthetical units are removed from all the sentences. Then, the sentences are clustered according to their similarity. Larger clusters represent information that is repeated more often across input documents, hence the size of a cluster is indicative of the importance of that information. One representative sentence is selected from each cluster. These sentences are included in the summary in decreasing order of their cluster size. When evaluating simplified summaries over the corpus of DUC 2003 and 2004, *ROUGE-L* metrics were obtained for both summaries without simplification (0.3643) and with simplification (0.3839). As the results have shown, the latter proved to be better summaries.

Also, Zajic et al. (2007) uses sentence compression as a preprocessing step of a multi-document summarization system. Sentence compression is used to generate headlines for news stories. Firstly, the five initial sentences of each document are selected for compression. Then, a pool of candidate sentences for inclusion in the summary is created using all the possible compressed versions of each sentence, built by the sentence compression procedure. Subsequently, a sentence selector constructs the summary by iteratively selecting the candidates based on a linear combination of features until the desired summary length has been reached. Features include: sentence position, sentence and document relevance, sentence centrality, scores from the compression modules, a redundancy value, and the number of sentences already selected from the document.

The sentence compression procedure was firstly developed as a headline generator for news stories (Dorr et al., 2003). In order to create a headline, the system compresses the main topic sentence by removing grammatical constituents iteratively from the sentence parse tree, using lexical and syntactical rules. Several linguistic phenomena are considered depending on the level of detail. At sentence level, preceding adjuncts, conjoined sentences and conjoined verb phrases are eliminated. At clause level, temporal and modifier expressions are removed. Finally, at noun phrase level, determiners and relative clauses are deducted. This work assumes that when creating a headline there is no need to create a grammatically correct sentence. Thus, all these structures are removed starting by the ones having lower content – determiners and time expressions – and iteratively shortening all the other ones until the sentence is shorter than a given threshold. This system was evaluated by a human judge who rated it with 3.72 in a scale from 1 to 5.

Formally represented approaches

Formal representations of the text can be used to perform simplification. Klebanov et al. (2004) addressed text simplification, by developing the formal concept of easy access sentences (EASes). EASes sentences satisfy some requirements, such as (1) is a grammatical sentence; (2) has one finite verb; (3) does not make any claims that were not present, explicitly or implicitly, in the input text; and (4) the more named entities it contains the better it is. In order to construct EASes automatically from input text, a simplification algorithm, whose first goal was to improve the performance of information retrieval, was developed. This algorithm addresses several linguistic issues, which comprise, for instance, the resolution of pronouns and anaphoric references, the assignment of correct tense to verbs that depend on governing verbs or other elements, the decision of the implicit subject of the verb relative clauses, etc.

EASes are constructed in two steps. The first step aims to identify person names, using BBN's Identifier (Bikel et al., 1999), and dependency structures of the input text derived using MINIPAR (Lin, 1998). The second step constructs the final EASes by transforming verb tenses, identifying verb dependents, and getting more named entities. EASes bring related dispersed information, which can be useful in information retrieval applications.

Another approach is presented by Filippova (2010). In order to remove redundancy in the context of an extractive multi-document summarizer, Filippova presents a method based on word graphs. This method seeks to create a succinct sentence, that is a compressed version of sets of sentences that were identified as possibly bearing some redundancy. Considering a cluster of similar sentences, the procedure creates a new compressed sentence that represents the cluster.

Considering all the words in the sentences of the cluster, a directed word graph is built by linking word A to word B through an adjacency relation. A word is mapped into a node provided that it is the same lowercased word and it has the same part-of-speech tag. Nodes are connected considering the sequence of each sentence. Their weights increase each time a similar connection is found in a new sentence. After the graph has been created, the shortest path between the beginning of a sentence and its end must be found. This path is likely to mention salient words from the input and put together words found next to each other in many sentences. Paths without verbs – as sentences without verbs are not valid sentences – and with less than eight words – shorter paths would

produce incomplete sentences – are ignored. The path with the minimum total weight is selected as the summary for that cluster. This method was used to address one of the major challenges of multi-document summarization, avoiding redundancy in the summaries produced. The quality of this method was assessed by human users, that, when run over a corpus of English texts, rated the system with an informativeness score of 1.30 and a grammaticality score of 1.44, in a scale from 0 to 2.

In this same research line, Lloret (2011) proposed a text summarization system that combines textual entailment techniques, to detect and remove information, with term frequency metrics to identify the main topics in the collection of texts. A word graph method is used to compress and fuse information, in order to produce abstract summaries. In the same way as Filippova (2010), the shortest path in a directed weighted word graph is identified. As the graph accounts for all the combinations of new sentences, the path defines which ones should be selected to include in the summary. Prior to this, a set of rules is applied to filter the number of incorrect generated sentences. These rules define that sentences must have a minimal length (three words), must contain a verb, and should not end in some specific words – articles, prepositions, conjunctions, and interrogative pronouns. Finally, the sentences that compose the final summary are selected up to a desired length or compression rate. An f-measure of 0.395 obtained with *ROUGE-1* was reported for the complete summarization procedure.

A more recent approach uses semantic information to perform sentence compression (Yoshikawa et al., 2012). These constraints are based on semantic roles in order to directly capture the relations between a predicate and its arguments. They claim that lexical and syntactic information may not be enough to perform compression since it does not differentiate specific expressions that when taking into account semantic annotation would be identified. Thus, semantic constraints allow for a fine-grained removal, improving sentence compression by selecting correct predicate-argument structures.

Machine learning approaches

Some approaches use machine learning techniques to perform simplification.

Cohn and Lapata (2007) deal with text simplification using a synchronous tree substitution grammar, a formalism that allows local distortion of the tree topology and can naturally capture structural mismatches. Through a grammar, which was induced from

a parallel corpus of uncompressed and compressed sentences, these authors propose a weighted tree-to-tree transducer abstract model that is used for text rewriting tasks. In a later work (Cohn and Lapata, 2009), they experimented that model in the sentence compression task, and they were able to improve state-of-the-art results, in terms of deletion tasks. The model not only is not deletion-specific, but mainly accounts for ample rewrite operations and scales to other rewriting tasks. They developed an algorithm – based in the large margin algorithm (Tsochantaridis et al., 2005), which efficiently learns a prediction function to minimize a given loss function –, that can be used in learning the model weights and mainly decoding the most plausible compression under the model. The quality of this approach has been assessed by 22 native English speakers that evaluated 30 simplified sentences. The results reported are an average of 3.38 points (in a 1-5 scale) considering the grammaticality of the simplified sentences and 2.85 points regarding the preservation of the most important information in the sentences.

Berg-Kirkpatrick et al. (2011) report a method that combines multi-document summarization with sentence compression. They learn a joint model that scores candidate summaries according to a combined linear model. This linear model is based in two features: (1) the n-gram types in the summary and (2) the compressions used. This approach outperforms unlearned baselines and the results achieved are highest than the ones published to date on the dataset of the Text Analysis Conference (TAC)[8] from 2008.

2.2.3 Simplification in Portuguese

Concerning the Portuguese language, SIMPLIFICA was the very first work on text simplification. Candido Jr. et al. (2009) addressed text simplification to promote digital inclusion and accessibility for people with low literacy levels. This system has been made available as a web service.

It encloses two modules: a statistical module (Gasperin et al., 2009b) in which a binary classifier decides whether a given sentence must or must not be simplified, and a rule-based module (Aluísio et al., 2008) which, based on the result of the classifier, executes or not the simplification operations.

In the statistical module, a binary classifier was trained by resorting to a parallel cor-

[8]The Text Analysis Conference (TAC – `http://www.nist.gov/tac/`) is a series of evaluation workshops that includes a Summarization track, the former DUC.

pus of news texts and their simplified versions, that were manually produced by an expert. The machine learning process decides whether a sentence must or must not be simplified. If the classifier decides in favor of simplification, the rule-based module is run. This module is based on a set of syntactic rules that define operations over certain linguistic phenomena, as appositions, relative clauses, coordinate or subordinate clauses, sentences with non-finite verbs, and passive voice sentences. Considering these phenomena, simplification operations, such as splitting sentences, changing a discourse marker by a simpler and/or more frequent one, changing passive to active voice, inverting clause order and non-simplification, define the syntactic rules.

An assessment of the performance of this approach in identifying the linguistic phenomena and in recommending the correct simplification is reported in (Gasperin et al., 2009a), when running the system over a manually built corpus of simplified news articles. An f-measure of 57.62 is achieved when performing all the operations proposed, while an f-measure of 58.61 is achieve when no operation is performed. These results suggest that blindly performing simplification by executing all the operations at the same time attains worse results than not simplifying at all. Although, when looking at the results of executing each operation by itself, "splitting sentences" achieved the best f-measure value (72.17), and "transformation to active voice" (44.14) and "inversion of clause ordering" (16.97) obtained the worst results, suggesting that "splitting sentences" can be the best approach to sentence simplification in Portuguese.

2.2.4 Summary

Text simplification is a recent natural language processing research area. The most common approaches to text simplification rely on induction of rules from annotated corpora, or on manual development of syntactic rules that define what to simplify. Still, most of the work done so far has been focused on syntactic transformations at sentence level, requiring relatively few linguistic knowledge, while achieving interesting results. However, there is still much room for improvement mainly when considering the discourse-level issues that arise from applying syntactic transformations, as those have not been a matter of research, specially when considering sentence reduction.

2.3 Discourse relations

Otterbacher et al. (2002) argue that revision strategies aiming to repair the problems found in multi-document summaries need discourse structure to be effective. Webber and Joshi (2012) defend also that discourse can enable language technology to overcome known obstacles to enhance performance. As stated in Chapter 1, in the context of improving the quality of a summary by running post-processing procedures over it, the present work adheres to this path of research. Discourse information is used as a means to improve text cohesion.

Firstly, some introductory definitions will be presented and the most common discourse theories used in the automatic processing of discourse relations will be detailed. Then, approaches aiming to find discourse relations are described, along with their evaluation metrics (whenever available).

2.3.1 Definitions

Ramsay defines discourse as "an extended sequence of sentences produced by one or more people with the aim of conveying or exchanging information" (Ramsay, 2004). The interconnections between sentences must follow some structure to ensure that the discourse is clear and can be fully understood. **Cohesion** is the property of a text that resorts on these connections.

The present work seeks to address cohesion when automatically generating texts. According to Halliday and Hasan (1976), cohesion is a surface level feature, and it deals with the relationships between text units. A process that ensures a linguistic relation between text segments is an element of cohesion.

Connectives are considered some of such elements. **Discourse connectives** are words or expressions that link discourse segments. The connection that they signal can be a local one, when the segments are sentences, or more global within the text, when the segments are paragraphs. Thus, when explicitly expressed, connectives realize discourse relations (Prasad et al., 2008).

Discourse relations define connections between discourse segments, while representing a sequential and hierarchical structure in a text. These relations can either be explicit or implicit. **Explicit** discourse relations are typically expressed using discourse connectives, that mark the specific type of relation those segments share. Yet, **implicit**

discourse relations need to be inferred by the reader.

As stated by Prasad et al. (2008), the text segments that express a discourse relation are typically named **ARG**$_1$ and **ARG**$_2$, where ARG$_1$ is the first segment, and ARG$_2$ is the second one, containing the discourse connective.

Consider the sentences in Examples 2.2 and 2.3, where, respectively, explicit and implicit discourse relations between sentences are illustrated.

Example 2.2

Sofrerá, provavelmente, uma sanção judicial, ***mas*** *estará em segurança.*
ARG$_1$=*(Sofrerá, provavelmente, uma sanção judicial),* ARG$_2$=*(****mas*** *estará em segurança.)*
ARG$_1$=He will, probably, suffer a judicial sanction, ARG$_2$=**but** he will be safe.

ARG$_1$ – *Se tivéssemos ganho já não havia jornais à venda.*
If we had won there was no newspapers for sale.
ARG$_2$ – ***Assim****, está ali quase tudo.*
Thus, there is almost everything.

Example 2.3

ARG$_1$ – *Não tenho dúvidas de que irá regressar.*
I have no doubt that he is returning.
ARG$_2$ – *(*IMPLICIT=***porque****) Faz falta ao futebol.*
(IMPLICIT=**because**) Football lacks him.

ARG$_1$ – *Pelo que fez na segunda parte, o Barcelona mereceu ganhar.*
For what they did, Barcelona deserved to win.
ARG$_2$ – *(*IMPLICIT=***mas****) Não vamos desanimar, o Barça e o Real ainda vão perder pontos.*
(IMPLICIT=**but**) Let us not be discouraged, Barça and Real will still lose points.

Text segments in Example 2.2 share, respectively, a CONTRAST and a CAUSE relation[9] as these are the relations their respective connectives express. *Mas* ("but") explicitly defines a CONTRAST relation between the two segments in the first sentence. Considering the second segment, a CAUSE relation is expressed by the connective *assim* ("thus").

Otherwise, in Example 2.3, the text segments are not explicitly linked by an overt discourse connective. These segments share CAUSE and CONTRAST relations[10], respectively. As

[9] Considering the classification proposed in this work (cf. *Annex D.2.1*), the fully class of these relations would be COMPARISON-CONTRAST-OPPOSITION and CONTINGENCY-CAUSE-RESULT.

[10] Considering the classification proposed in this work (cf. *Annex D.2.1*), the fully class of these relations would be CONTINGENCY-CAUSE-REASON and COMPARISON-CONTRAST-OPPOSITION.

they have no overt marker, these relations are said to be implicit. Despite this, as stated in the example, a connective can be inferred taking into account the meaning of the sentences and the information they convey.

2.3.2 Discourse theories

In order to map this sort of discourse relations into a formal structure, several discourse theories were proposed.

Rhetorical Structure Theory (*RST*) (Mann and Thompson, 1998) is a discourse theory seeking to formally represent a text. *RST* assumes that a text may be structured as a discourse tree, whose intermediate nodes are discourse relations and leaves are propositional units expressed by segments (usually clauses) in the text. It is intended to describe texts, rather than the processes of creating or reading and understanding them. The central notion in *RST* is the rhetorical relation, which defines a connection between two non-overlapping text units, the nucleus and the satellite. The nucleus conveys the most essential information for the speaker's purpose. It is independent from the satellite, while the satellite can be dependent from the nucleus.

The structure of the text arises from a set of constraints that are associated with each relation. These constraints operate on the nucleus, on the satellite, and on the combination of both. Rhetorical relations are then assembled in *RST*-trees, based on a specific constituency schema.

While *RST* relates segments of the discourse that are in a single document, *Cross-document Structure Theory* (*CST*) (Radev, 2000) was developed to analyze a collection of related documents, in the context of multi-document summarization. Other than *RST*, this theory makes no assumptions about the authors' intentions in creating cohesion in texts. Rather, *CST* is a theory seeking to describe the semantic connections among units of related documents, defining a structure between those documents. It assigns labels to cross-document conceptual links that state the type of relation the documents hold.

This model proposes a set of 24 relations that make explicit the relations among different parts of the texts. This set includes relations common to the multi-document phenomena, such as contradiction, redundancy or complementarity.

CST structure is represented through a graph where the nodes are text units (either words, phrases, sentences, paragraphs or even documents), and the edges are *CST*-relations.

Other theories were built based on specific problems to be solved. Lascarides and Asher (2007) built the *Segmented Discourse Representation Theory* (*SDRT*), based on the formal theory, *Discourse Representation Theory* (*DRT*), developed by Kamp and Reyle (1993). The idea behind *DRT* is that any discourse in natural language can be interpreted in the context of a semantic representation structure. The formulation of *DRT* interpretation involves two stages: (1) the construction of semantic representations of the input, named DRSs (Discourse Representation Structures), said to be an abstraction of the mental representation of discourse; and (2) a set-theoretic interpretation of those DRSs.

DRSs are built up by the hearer as the discourse unfolds, and are composed by two parts: the discourse referents, representing the objects under discussion, and a set of DRS-conditions that encode the information accumulated on these discourse referents.

Every new piece of discourse is interpreted against and in turn updates the representation of the already processed discourse. This way, this theory broke with the classical formal semantics (Montague, 1974), as it relies in an interpretation of the discourse, other than of individual sentences.

SDRT is an extension of *DRT*, that assigns rhetorical relations with precise dynamic semantics. It extends the information generated by the grammar with commonsense reasoning including both linguistic and non-linguistic information, in order to create a more complete semantic representation of the discourse. In summary, it enriches logical forms to include rhetorical relations, to which semantic information is assigned.

2.3.3 Approaches

The intent of the majority of the studies that address discourse relations is to recognize [(Marcu and Echihabi, 2002), (Lapata and Lascarides, 2004), (Blair-Goldensohn et al., 2007), (Lin et al., 2009), (Park and Cardie, 2012), (Versley, 2013)] and classify [(Wellner et al., 2006), (Pitler et al., 2008), (Pitler and Nenkova, 2009)] discourse relations in unseen data.

Other works [(Louis et al., 2010), (Biran and Rambow, 2011), (Feng and Hirst, 2012)] approach this problem with different goals. Louis et al. (2010) aim to enhance content selection in single-document summarization. Biran and Rambow (2011) use discourse relations to detect justifications of claims made in written dialog. Feng and Hirst (2012) seek to improve the performance of a discourse parser.

Despite of their different goals, these studies follow a common approach to find and classify discourse relations in text, that is a machine learning approach. First, a corpus annotated with discourse relations is selected. Afterwards, a set of features are identified in order to train a classifier that learns how to distinguish discourse relations. Finally, the model is used to classify discourse relations in raw text.

The remainder of this Section is organized based on the structure of a machine learning approach. The discourse corpora that have been made available are described in Section 2.3.3.1. The features, the classifiers and their evaluation are reported Section 2.3.3.2.

2.3.3.1 Corpus

Most of the work concerning discourse relations uses corpora annotated by hand. There are some discourse corpora available, that were built using different paradigms. The *Penn Discourse TreeBank* (*PDTB*) ((Prasad et al., 2007) and (Prasad et al., 2008)) is a large English corpus, annotated by human experts with information related to discourse structure and discourse semantics. This corpus supports the extraction of syntactic and semantic features, providing novel information that aids in the development of practical algorithms. Discourse relations are defined by explicit phrases or by structural adjacency. Each relation is composed by two arguments, annotated with the sense of the relation and the attributions associated with the relation and each of its arguments. It is hierarchically organized relying in four top classes: TEMPORAL, CONTINTENCY, COMPARISON and EXPANSION. Annotations in the *PDTB* are aligned with the syntactic constituency annotations of the *Penn Treebank*.

Also, Wolf and Gibson (2005) proposed a graph-based approach to represent informational discourse relations, that contain a small number of discourse relations, creating a more generalizable representation of the discourse structure. These relations hold between sentences or other non-overlapping segments in a discourse, and follow the structure defined in Hobbs (1985). They are represented by conjunctions that connect the segments resulting in passages that evidence the coherence relation. Using this approach, a corpus, the *Discourse GraphBank*, manually annotated with those coherence relations, has been created.

Carlson et al. (2001) developed a corpus based in *RST*. The *RST Treebank* contains discourse tree annotations for news articles from the *Penn Treebank*. This corpus is grounded

in a theoretical approach (*RST*) and is large enough to offer wide-scale use, including linguistic analysis, training of statistical models of discourse, and other computational linguistic applications. Human annotators analyzed documents and build the corresponding discourse tree, based on *RST* discourse relations.

Finally, Radev et al. (2004) created the *CSTBank*, a corpus of multi-document clusters, in which pairs of sentences from different documents have been annotated with *CST* relationships. Firstly, clusters of related news articles were created. Then, human annotators marked *CST* relationships across the document clusters. This corpus was built to aid the automatic identification of *CST* relationships in arbitrary clusters of related articles.

2.3.3.2 Features, classifiers and evaluation

In order to identify the most relevant features that can distinguish the discourse relations that two text segments share, several experiments were performed over the previously mentioned corpora.

Word pairs has been the most common feature used. When defining word pairs, several have been the combinations experimented.

The pioneer work by Marcu and Echihabi (2002) inspired the research on finding discourse relations. These authors describe an unsupervised approach to recognize specific discourse relations (CONTRAST, EXPLANATION-EVIDENCE, CONDITION and ELABORATION), that hold between two spans of texts. Firstly, they started by creating a training corpus by extracting adjacent sentence pairs that contain specific connectives for each discourse relation in question (for instance, they used "but" for CONTRAST relations). In addition, sentences containing a connective in the middle are extracted. Then, they split each extracted sentence into two spans, one containing the words from the beginning of the sentence to the occurrence of the connective (e.g. the word "but") and one containing the words from the occurrence of the connective to the end of the sentence. Finally, all the pairs are labeled with the relation expressed by the connective.

In order to define their feature set, they hypothesize that lexical item pairs provide clues about the discourse relations shared by the text spans in which those items occur. So, they extract an opposite polarity word (nouns, verbs, and other cue-phrases) from each sentence of the pairs previously gathered. These pairs of words were used as features of a Bayesian classifier, that learns the discourse relation between them. Further ex-

periments shown that the training dataset provided to the classifier contained too much noise. Thus, instead of simply using the sentence pairs, they used the most representative words (nouns, verbs and cue-phrases) from each pair. The results with over 75% of accuracy for all the classes suggested that discourse relation classifiers trained on most representative word pairs and millions of training data can achieve high performance.

Blair-Goldensohn et al. (2007) proposed several refinements to the word pair model focusing on two specific discourse relations: CAUSE and CONDITION. The refinements proposed include (1) stemming, (2) using a small fixed vocabulary size consisting in the most frequent stems, and (3) a cutoff on the minimum frequency of a feature. Blair-Goldensohn et al. (2007) reproduced in their corpus the approach by Marcu and Echihabi (2002) in order to compare both approaches. They report that Marcu and Echihabi's system achieved a 54% accuracy, while their own system achieved 60% accuracy.

As opposed to these experiments, a study by Pitler et al. (2008) claims that word pairs lead to data sparsity problems. A possible approach to solve this problems is to reduce the number of features, using semantic relations between words. Based on these conclusions, these authors tested other features as polarity tags, length of verb phrases, verb tenses, context windows in the arguments, and word pairs. In these experiments, only the top classes of *PDTB* were used. Nevertheless, some interesting conclusions on the best features were drawn. They found that word polarity, verb classes as well as some lexical features are strong indicators of the type of discourse relation. *Naïve Bayes* and *Maximum Entropy* were the classifiers used and *Naïve Bayes* produced the best results, a 94% of accuracy when classifying explicit discourse relations.

In a similar vein, Wellner et al. (2006) seek not only to identify but also to classify discourse relations, using the *Discourse GraphBank*. A variety of syntactic and lexical-semantic features were experimented, and most of them were obtained by using external knowledge-sources. The set of features tested included (i) words appearing in the beginning and end of the two discourse segments; (ii) proximity between two segments and their direction; (iii) paths in an ontology between non-function words occurring in the two segments; (iv) word-pair similarities between words; (v) grammatical dependency relations between the two segments; and (vi) event-based attributes (e.g. cue-words from one segment paired with events in the other segment). Context words (i) and proximity (ii) were the most impacting features, when considered alone. In addition, several combination of these features were performed. There is a strong evidence that combined

features are responsible for a better performance of the classifier. They used a *Maximum Entropy Classifier* to assign labels to each pair of discourse segments connected by some relation. They report accuracy results of over 70% when combining all the features.

Lin et al. (2009) used four classes of features: contextual features, constituent parse features, dependency parse features, and lexical features. In order to define contextual features, they rely on the definition of the dependencies that hold between pairs of discourse relations stated by Lee et al. (2006). By observing the *PDTB* looking for those dependencies, they found that from the set defined by Lee et al. (2006), two of them – embedded argument and shared argument – are the most common ones, so those were used as context features. Constituent parse features resort to syntactic structure, as syntactic structure within one argument may constrain the relation type and the syntactic structure of the other argument. So, the production rules of the constituency trees of the two arguments are used as features. Dependency parse trees are also used, due to the fact that those encode additional information at the word level that is not explicitly present in the constituency trees. Finally, in line with Marcu and Echihabi (2002), word pairs are also used as lexical features. Using a *Maximum Entropy* classifier, they reported an accuracy of 40% when applying all features classes, an improvement of 14.1% over the baseline.

Louis et al. (2010) experimented three types of features: structural, semantic, and non-discourse features. Structural features were based on *RST* trees and include the depth of the text span and its promotion score (text spans considered more salient are awarded with promotion scores). Semantic features, obtained from *PDTB*, encode the number of relations (implicit, explicit and total) shared by a sentence and the distance between the arguments of those relations. Finally, non-discourse features are standard features used for summarization, such as (1) length of the sentence; (2) if the sentence is the first sentence of the paragraph or the first sentence of a document; (3) the sentence offset from document beginning, as well as paragraph beginning and end; (4) the average, sum and product probabilities of the content words appearing in sentences; and (5) the number of topic words in the sentence. The results obtained provide strong evidence that discourse structure, other than discourse semantics, is the most useful aspect in improving sentence selection for summarization. Still, both these types of discourse features complement the commonly used non-discourse features for content selection. Accuracies over 75% were obtained by the classifiers, and the single-document summarization system improved its performance when tested with different features.

2.3.4 Summary

Typical approaches to find and classify discourse relations rely on machine learning procedures. Firstly, a corpus that relates text segments in a discourse manner is needed. In addition, a set of feature classes that represent the way those text segments are related must be defined. Finally, based on a training dataset relying on those features, a classification algorithm learns how to assign a discourse relation to two unseen text segments.

Conclusions from the experiments reported in this Section can be drawn. Regardless of the corpus used, word pair features (nouns and verbs), specially their lemmas, are the ones that produce the best classification results. Also, linguistic features (when available) aid to improve the classification procedure. Bayesian classifiers prove to produce very good results when classifying discourse relations between two texts spans.

Far from being perfect, the results achieved can still be used with very interesting outputs.

CHAPTER 3

SUMMARIZATION

SIMBA is an automatic multi-document summarization system that deals with texts in Portuguese. This system was built by us to validate the hypotheses stated in Section 1.4.

This system includes two modules. The first module aims to perform extractive summarization of a collection of documents written in Portuguese. The second module seeks to improve the textual quality of the summary produced by the first module by seeking to transform it into a more fluent and readable text.

In order to build SIMBA, most of the decisions, which reflect the main challenges addressed by the system, took into account the conclusions drawn after a set of manual experiments, performed by language experts (Silveira and Branco, 2012). Yet, when considering the summarization procedure, other options reflect frequently used methods that proved to be effective in systems covering other languages as well.

This Chapter describes the summarization process (the first module). The summarization process aims to combine the best strategies found in the literature (described in Section 2.1) in order to improve the selection of the sentences that will figure out in the summary. At the same time, it discusses the decisions pursued to improve the performance of the system.

3.1 Overview

In order to work towards validating the hypotheses raised (see Section 1.4), several issues must be taken into account when constructing a summarization system.

First, it is important to define the goals and features of the system. Then, it is necessary to define strategies in order to achieve these goals, based on these features.

The main design features of the system are described in Table 3.1:

Input features	language	Portuguese
	span	multi-document
	genre	general
Purpose features	audience	generic
	function	informative
Output features	length	compression rate defined by the user
	relation to source	extract

Table 3.1: Summarizer design features

The goals of the system reflect these design features. As the very first goal of a summarization system, that aims to produce generic and informative summaries should be to **select the most relevant content in the collection of documents**, when it comes to multi-document summarization, the system must also **discard redundant information collected in different documents**. Moreover, as the content of the summary can be retrieved from different sources, **organizing that information to form a meaningful text** is another goal of such a system. Finally, due to the extractive nature of the summary, in order to **create a readable text to be used by human readers**, SIMBA includes a post-processing procedure that, taking into account the compression rate previously defined, seeks to improve the textual quality of the final summary.

Once the design features have been defined, the strategies to accomplish the goals of this system were determined, and will be detailed in the following sections, through the description of the typical stages of a summarization system.

The main stages of a multi-document summarization system comprise typically five stages, that can be also found in the process of single-document summarization (cf. Section 2.1.1), as defined by Mani (2001a).

SIMBA performs summarization based on these general phases, described in the following sections:

- Section 3.2 details the Annotation phase (Analysis in Mani's taxonomy), where all the documents are prepared to be summarized;
- Section 3.3 describes the Content Selection phase (Transformation), where the sentences that compose the summary are identified;
- Section 3.4 reports the Summary Generation phase (Synthesis), where the final text to be delivered to the user is created.

The basic procedure is depicted in Figure 3.1, and each one of its phases will be described in detail below.

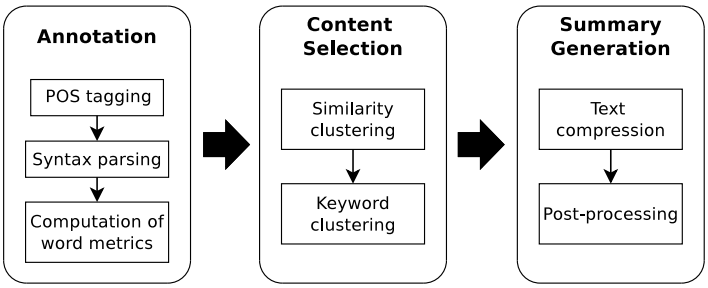


Figure 3.1: SIMBA architecture overview.

Annotation takes a collection of documents as input and annotates their texts using language processing tools specifically built to deal with Portuguese. **Content Selection** identifies the most relevant pieces of information in this collection through a double clustering approach, tackling two specific challenges of multi-document summarization: redundancy removal and relevant information selection. Finally, **Summary Generation** creates the final text by performing specific tasks that seek to improve the fluency and readability of the final summary. The present Chapter reports on most of the architecture depicted above, except for the Post-processing procedure, included in the Summary Generation phase, that will be described in detail in Chapter 4.

Consider the texts in Examples 3.1 and 3.2, that will be used to illustrate the summarization process[1]. The texts were retrieved from the corpus used to evaluate this system

[1]The numbers in brackets are not part of the text. Those represent the sentence absolute positions for further reference. The paragraphs in the original texts are also represented.

(cf. Section 5.1). A translated version of these texts can be found in *Annex C.1*. These texts were submitted to SIMBA to be summarized. In the next sections they will be used as a working example to illustrate the different phases of the summarization procedure.

Example 3.1: Text #1

(1) Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.

(2) As vítimas do acidente foram 14 passageiros e três membros da tripulação. (3) Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu. (4) Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.

(5) O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala. (6) "Não houve sobreviventes", disse Okala. (7) O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.

Example 3.2: Text #2

(1) Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas. (2) As vítimas do acidente foram 14 passageiros e três membros da tripulação. (3) Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 Km do aeroporto de Bukavu.

(4) O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala. (5) "Não houve sobreviventes", disse Okala.

(6) O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.

(7) Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.

This working example is very interesting in many ways. First, note that Text #1 and Text #2 are very similar. Their sentences have many similarities and in some cases are even identical. In addition, the order of the sentences is very different, referring to different relevance assigned by each author to the information of the same sentence. Note, for instance, that sentence #4 in Text #1 is the last sentence (#7) in Text #2. Finally, the para-

graphs defined in both texts are different, as different sentences are grouped together either to emphasize different topics or to unify sentences considered to be related.

All in all, two texts with similar individual sentences can result in different texts.

3.2 Annotation

In the annotation phase, the collection of documents is prepared to be automatically processed. It consists in three steps: (1) document annotation, (2) sentence annotation, and (3) computation of word metrics.

The first step aims to annotate each document using *LX-Suite* (for its detailed description see *Annex B.1*). Sentence and paragraph boundaries are identified and words are tagged with their corresponding part-of-speech (POS) and lemma, by considering the context of their occurrence. Example 3.3 shows an annotated sentence.

Example 3.3

"Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa."
According to airport sources, the crew members were of Russian nationality.
<s>
Segundo/PREP fontes/FONTE/CN#FP aeroportuárias/AEROPORTUÁRIO/ADJ#FP ,*//PNT os/DA#MP membros/MEMBRO/CN#MP de_/PREP a/DA#FS tripulação/TRIPULAÇÃO/CN#FS eram/SER/V#II-3P de/PREP nacionalidade/NACIONALIDADE/CN#FS russa/RUSSO/ADJ#FS ./PNT
</s>

The main interest in using this tool consists in POS tagging and nominal and verbal lemmatization, that aid in the comprehension of the content of each sentence. Even so, it is important to keep two versions of the sentence: the original one and the annotated one. The original version of the sentence is used in the sentence reduction process of the summary generation phase (cf. Section 4.2.1). The annotated version of the sentence is used in the current step, when sentence scores are being computed, since both POS tags and word lemmas are considered when comparing sentence words. Thus, the information contained in both versions (original and annotated) is mapped into a single representation to handle differences due to the contraction of prepositions with other words, punctuation tokens, etc. The sentence in Example 3.3 was split in words and Example 3.4 shows the corresponding information.

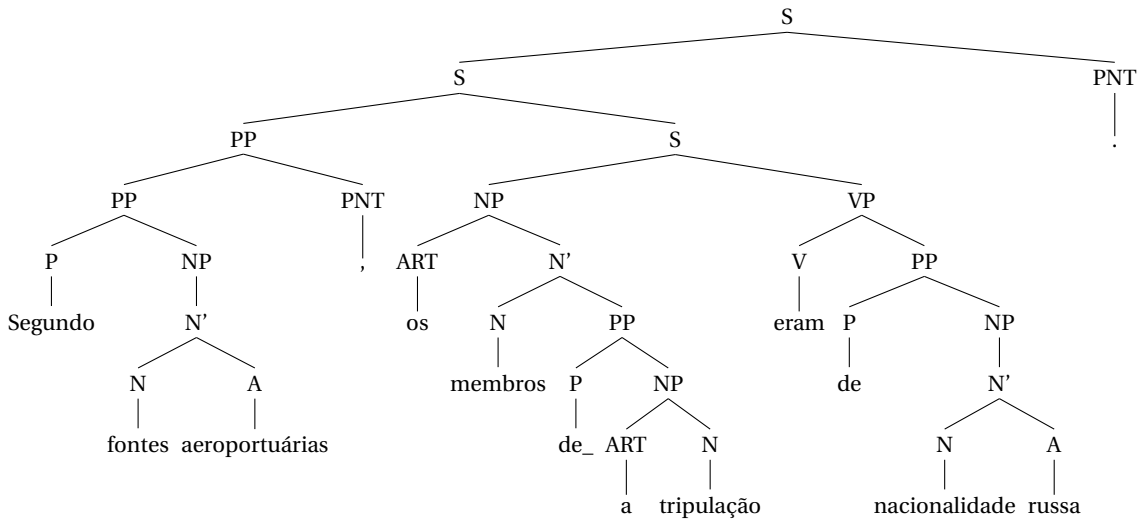
Example 3.4

Original	Tokens	Lemma	POS Tag
Segundo	Segundo		PREP
fontes	fontes	fonte	CN#FP
aeroportuárias,	aeroportuárias	aeroportuário	ADJ#FP
os	os		DA#MP
membros	membros	membro	CN#MP
da	de_a		PREP DA#FS
tripulação	tripulação	tripulação	CN#FS
eram	eram	ser	V#II-3P
de	de		PREP
nacionalidade	nacionalidade	nacionalidade	CN#FS
russa.	russa	russo	ADJ#FS

The second step of the annotation process parses every sentence with *LX-Parser* (described in *Annex B.2*). *LX-Parser* builds a parse tree (cf. Example 3.5), that represents the syntactic structure of the sentence. This parse tree will be used in the sentence reduction phase (see Section 4.2.1), where specific syntactic structures are identified and removed, if appropriate.

In the third step, the metrics concerning all the words in all the texts are computed.

Example 3.5



Despite being simple and not requiring a deep analysis, term-weighting metrics have been proven to be effective in producing good summaries (Orăsan, 2009). SIMBA seeks to use an effective, yet efficient method to identify the relevant information contained in a collection of documents, focusing mainly on the quality of the final text, that is delivered to the end-user. Thus, in order to streamline the summarization process, *tf-idf* is used as a ranking metric. Two main reasons drove the selection of *tf-idf* as a ranking metric: (1) it is a simple and efficient metric in identifying word relevance in any context or language and (2) it is very effective in filtering *stop-words*.

tf-idf reflects the importance of a word to a document contained in a collection of documents, and this is the reason why it is very suitable for multi-document summarization. Moreover, not every word in a text is a relevant word. Despite being very frequent, there are some words, commonly named *stop-words*, that are not to be considered relevant for the purpose at stake. Most NLP systems deal with *stop-words* by creating a list of words to ignore. Taking into account the requirement of language-independence for our system, to build and use such a list would not be an option. Considering all this, by matching all the requirements of the algorithm, *tf-idf* was the metric selected, being computed as described in Equation 3.1.

$$tf_t = \frac{n_t}{\sum_{w \in D} n_w} \qquad idf_t = \log \frac{N}{1+d_t}$$

$$\text{tf-idf}_t = tf_t \times idf_t \tag{3.1}$$

where

tf_t	–	frequency of word t.	n_w	–	number of occurrences of word w.
idf_t	–	inverse document frequency.	N	–	number of documents.
n_t	–	number of occurrences of word t.	d_t	–	number of documents where the word t occurs.
D	–	document collection.			

When computing *tf-idf*, word occurrences are computed considering the lemmas. As *tf-idf* reflects the relevance of a word, another metric is needed to define the relevance of a sentence. The most straightforward approach would be to sum the *tf-idf* scores of the words composing a sentence, so that sentences containing the words with the highest *tf-idf* scores are considered the most significant ones. In SIMBA, along with this ***main score***

– the sum of the *tf-idf* scores of each sentence divided by the number of words in the sentence – another score dimension complements this one. This dimension is named ***extra score*** (further discussion about the *extra score* can be found in Section 3.3). This score is used in the summarization process to reward or penalize the sentences, by adding or removing predefined score values.[2] This way, the relevance of a sentence is given by the combination of two metrics, one that depends on the *tf-idf* scores of the words (*main score*), and another one (*extra score*) depending on the content selection phase (cf. Section 3.3). The ***relevance score*** is the sum of these two metrics, as shown in Equation 3.2.

$$mainScore_s = \frac{\sum_{w \in s} tf\text{-}idf_w}{n_s}$$

$$relevanceScore_s = mainScore_s + extraScore_s \tag{3.2}$$

where

w	–	word.
s	–	sentence.
n_s	–	number of words in sentence s.
$tf\text{-}idf_t$	–	term frequency – inverse document frequency of the word w.

Once the word properties have been computed, all the relevant statistical information about the texts have been gathered and can now be used to select relevant data. Thus, this stage provides an important tool for the remainder of the summarization process, since it defines the basis of the extraction procedure.

3.3 Content selection

Content Selection aims to identify relevant information in the collection of texts.

An extraction-based summarization system takes the sentences from the collection of texts submitted as input in order to create a summary. These texts, by addressing the same subject, may contain redundant information. As a summary consists in a shorter version of the original texts, only the most relevant content expressed in those texts should be retrieved to be part of the summary. The two main goals of multi-document summarization are then (1) removal of redundancy and (2) selection of the relevant content.

[2] The predefined value to be added to the extra scores is set to 0.5 as a reward or -0.5 as a penalty value.

In this work, these goals are tackled through a double clustering approach, which includes a **similarity clustering phase** (cf. Section 3.3.1) and a **keyword clustering phase** (cf. Section 3.3.2). This approach meets the requirements of both these goals by relying in a fast and efficient solution – clustering – to analyze and organize large quantities of unrelated information. Other studies addressed these goals [(Radev et al., 2000), (Lin and Hovy, 2000), (Farzindar et al., 2005)] by combining a relevance metric, such as *tf-idf*, with an overlapping metric, such as *cosine similarity*, but without defining any structure in the data.

The two clustering steps are executed in sequence, that is the input of the second step is the output of the first one. In the first step, redundancy is addressed by clustering similar sentences on the basis of a measure of similarity. In the second step, the sentences identified as non-redundant are assembled by topics, using the keywords retrieved from the collection of texts.

This approach has impact on the content of the output summary. On the one hand, the similarity clustering helps to ensure that its content is not repetitive. On the other hand, keyword clustering assures the selection of the most significant content, the preservation of the idea of the input texts, and the organization of the final summary.

3.3.1 Clustering sentences by similarity

Similar sentences convey similar information, being redundant among themselves. This task aims thus to find a similarity relation between a set of unorganized sentences. At this point, it is important to find groups of sentences with the same degree of similarity.

Consider, for instance, the collection of sentences in Table C.1 (in *Annex C*) retrieved from the texts in Examples 3.1 and 3.2, and ordered by their *relevance score* (introduced in Section 3.2). Recall that the *relevance score* is the sum of the *main score* and the *extra score*. At this point, the *relevance score* is equal to the *main score*, as no specific summarization procedure has been accomplished, since the *extra score* has not been updated yet.

Suppose that we want to build a summary using this collection of sentences, without further processing, with a compression rate of 20%. This means that from the total words in the input texts (252), the summary should have 50 words. Taking into account the ordered list of sentences in Table C.1, by retrieving 20% of the words from it, we would include in the summary the sentences shown in Example 3.6.

Example 3.6

S_1 *O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*
The spokesman said the plane, a Soviet Antonov-28 Ukrainian manufacturing and property of a company in Congo, the Congo Trasept also took a load of minerals.

S_2 *O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*
The spokesman said the plane, a Soviet Antonov-28 Ukrainian manufacturing and property of a company in Congo, the Congo Trasept also took a load of minerals.

These two sentences, by being the first ones in a list ordered by the current *relevance score*, are considered the most relevant sentences found in the collection of texts. However, these sentences are identical, which means that the summary would contain repeated information.

Hereupon, a division algorithm is applied to the complete set of sentences in order to group in the same cluster the sentences that are similar. A cluster is composed by a collection of sentences, a similarity score, and a representative sentence. The representative sentence is the sentence with the highest *main score*.

The clustering algorithm comprises the following steps:

1. Start with an empty set of clusters;
2. Create the first cluster with the first sentence of the collection;
3. Considering a next sentence in the collection of sentences:
    a) Compute the similarity score with the representative sentence of each cluster;
    b) Obtain the maximum similarity score;
    c) Compare this similarity score with the similarity value of each cluster;
        i. If the similarity score is above the threshold, adds the sentence to the current cluster;
        ii. If the similarity score is below the threshold, creates a new cluster with the current sentence;
4. Repeats Step 3 until all the sentences have been considered.

When adding a sentence to a cluster, the sentence that is the cluster representative must be updated. If the *main score* of the sentence being added is higher than the previous representative one, the newly added sentence becomes the representative of the cluster. Each representative is rewarded by updating its *extra score* (introduced in Section 3.2). In the same way, the *extra score* of the sentences that have been replaced as representatives is removed.

Considering the collection of sentences discussed above, the similarity clustering procedure creates the clusters shown in Table C.2 (in *Annex C*). From a collection of 14 sentences, the similarity clustering produces 7 clusters. The first sentence in each cluster is the representative one. Note that the representative sentences have a higher *relevance score*, than the other sentences in the cluster. This is due to the update of the *extra score* of each sentence that is defined as representative.

After the clustering procedure, if the sentences are ordered based on their *relevance score*, we would obtain the ordered list of sentences shown in Table C.3 (in *Annex C*). Regard that, as expected, the representative sentences ascended to the first positions in the list. This way, the repetitive information has been relegated to the rest of the list of candidate sentences, having a lower probability to be selected to be part of the final summary. Comparing to the order presented in Table C.1, this new order shows the relevance of using the *extra score* to reward the representative sentences and to penalize the sentences considered redundant.

After this, only the representative sentence of each cluster is considered, in order to feed the next clustering stage. Table C.4 shows the final set of sentences that is used as input in the clustering by keywords phase.

The similarity clustering process is depicted in Figure 3.2.

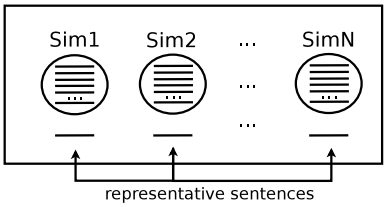


Figure 3.2: Similarity clustering.

Thus, a collection of sentences containing the representative sentences is built, defining the input of the next phase of the summarization process.

Similarity among two sentences

Though typically a single overlap metric has been used in the literature [(Radev et al., 2000), (White et al., 2001), (Farzindar et al., 2005)], we considered relevant to use a combined metric that not only accounts for in-sequence matches between the sentences, but also for all the matches between them.

Thus, the similarity between two sentences comprises two dimensions: the **word subsequences** and the **word overlap**. A subsequence is a sequence of common words between two sentences. Overlap is the number of words, in sequence or not, shared by two sentences. Both the word subsequences and the word overlap scores are computed considering the word lemmas. The scores for these two dimensions are combined in order to compute the **similarity score** among the two sentences, as shown in Equation 3.6.

$$sumSubsequences(s_1, s_2) = \sum_{i>1} \left(\frac{subsequence_i}{numberWords_{s_1}} + \frac{subsequence_i}{numberWords_{s_2}} \right) \tag{3.3}$$

$$subsequences(s_1, s_2) = \frac{sumSubsequences(s_1, s_2)}{numberSubsequences(s_1, s_2)} \tag{3.4}$$

$$overlap(s_1, s_2) = \frac{commonWords(s_1, s_2)}{totalWords_{s_1} + totalWords_{s_2} - commonWords(s_1, s_2)} \tag{3.5}$$

$$similarity(s_1, s_2) = \frac{subsequences(s_1, s_2) + overlap(s_1, s_2)}{2} \tag{3.6}$$

where

subsequences(s_1, s_2)	–	subsequences ratio between s_1 and s_2 – subsequence score.
sumSubsequences(s_1, s_2)	–	sum of the number of tokens of the subsequences between s_1 and s_2.
subsequence$_i$	–	number of tokens of the subsequence i.
numberSubsequences(s_1, s_2)	–	number of subsequences between s_1 and s_2.
overlap(s_1, s_2)	–	overlapping tokens between s_1 and s_2 – overlap score.
commonWords(s_1, s_2)	–	number of tokens common to s_1 and s_2.
numberWordss_i	–	number of tokens in the sentence i.

The **similarity score** is obtained as an average of the two dimensions that compose it. It indicates that two sentences are similar the more number of words in sequence they have and, at the same time, the more number of overlapping words they share. This score is confronted with a predefined threshold – similarity threshold (set to 0.75) – that

establishes the limit above which two sentences can be considered similar, for the present purposes. This value determines that the sentences must have at least a similarity score of 0.75. Thresholds from 0.50 to 0.90 were tested empirically. This value proved to be the one that maximizes the clustering creation, being it possible to distinguish the sentences that are to be grouped as being repetitive from the ones that are not.

The calculation of the similarity score is illustrated, using the sentences in Example 3.7 – retrieved from the two texts in Examples 3.1 (sentence #1) and 3.2 (sentence #1) –, firstly by computing the word subsequences shared by both sentences, then by computing their word overlap score, and finally by computing their similarity score.

Example 3.7

S_1 *Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*
A crash in the town of Bukavu, in the eastern Democratic Republic of Congo, killed 17 people on Thursday afternoon, said today a spokesman for the United Nations.

S_2 *Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.*
A crash in the town of Bukavu, in the eastern Democratic Republic of Congo (DRC), killed 17 people on Thursday afternoon, said on Friday a spokesman for the United Nations.

Example 3.8 shows all the subsequences common to both these sentences.

Example 3.8

S_1 *Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*
A crash in the town of Bukavu, in the eastern Democratic Republic of Congo, killed 17 people on Thursday afternoon, said today a spokesman for the United Nations.

S_2 *Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.*
A crash in the town of Bukavu, in the eastern Democratic Republic of Congo (DRC), killed 17 people on Thursday afternoon, said on Friday a spokesman for the United Nations.

The first subsequence (marked with a single underline) contains 14 words. The second subsequence (marked with a wave underline) contains 8 words. The third subsequence (marked with a double underline) contains 5 words. Note that subsequences con-

taining only one word are not considered, since those are represented by the word overlap score. Considering this, the subsequences overlap score between these two sentences is computed in Example 3.9.

Example 3.9

$$\begin{aligned} sumSubsequences(s_1, s_2) &= \sum_{i>1} \left(\frac{\text{subsequence}_i}{\text{numberWords}(s_1)} + \frac{\text{subsequence}_i}{\text{numberWords}(s_2)} \right) \\ &= \left(\frac{14}{28} + \frac{14}{30} \right) + \left(\frac{8}{28} + \frac{8}{30} \right) + \left(\frac{5}{28} + \frac{5}{30} \right) \\ &= 1.86 \end{aligned} \tag{3.7}$$

$$\begin{aligned} subsequences(s_1, s_2) &= \frac{\text{sumSubsequences}(s_1, s_2)}{\text{numberSubsequences}(s_1, s_2)} \\ &= \frac{1.86}{3} \\ &= \mathbf{0.621} \end{aligned} \tag{3.8}$$

The word overlap score has also been computed for the sentences in the previous example (Example 3.10). The subsequences overlap score (Equation 3.8) along with the word overlap score (Equation 3.9) define the final similarity score (Equation 3.10) for both sentences.

Example 3.10

$$\begin{aligned} overlap(s_1, s_2) &= \frac{\text{commonWords}(s_1, s_2)}{\text{totalWords}_{s_1} + \text{totalWords}_{s_2} - \text{commonWords}(s_1, s_2)} \\ &= \frac{27}{28 + 30 - 27} \\ &= \mathbf{0.871} \end{aligned} \tag{3.9}$$

$$\begin{aligned} similarity(s_1, s_2) &= \frac{\text{subsequences}(s_1, s_2) + \text{overlap}(s_1, s_2)}{2} \\ &= \frac{0.621 + 0.871}{2} \\ &= \mathbf{0.75} \end{aligned} \tag{3.10}$$

The **word overlap score** (detailed in Equation 3.5) is computed using the Jaccard index (Jaccard, 1908), a commonly used similarity measure, which consists in the number of overlapping words between two sentences, divided by the sum of the lengths of the two sentences subtracted by the number of common words in both sentences (since in the sum of both sentences, the overlapping words are counted twice).

When considering two identical sentences, as the ones in the Example 3.11, all these three values are the same and equal to 1.0, since the same words are in the same order and in the same sequence. Consequently, these sentences should be regarded as similar.

Example 3.11

Sentence#1:
O João ama a Maria.
John loves Mary.

Sentence#2:
O João ama a Maria.
John loves Mary.

Word Overlap	Subsequences Overlap	Similarity Score
1.0	1.0	**1.0** $\geq$ 0.75

In fact, the word overlap score alone does not reflect the similarity between sentences, since it would assign the same similarity score to sentences that have the same words in the same order and to sentences that have the same words but in different order.

Consider Example 3.12, where the two sentences despite sharing the same words cannot be considered similar.

Example 3.12

Sentence#1:
O João ama a Maria.
John loves Mary.

Sentence#2:
A Maria ama o João.
Mary loves John.

Word Overlap	Subsequences Overlap	Similarity Score
1.0	0.4	**0.7** $<$ 0.75

Example 3.12 shows that despite having the same words ($overlap$ = 1.0), the sentences have different meanings, thus should not be considered similar. By only using the word overlap score these sentences would be considered similar and would be included in the same cluster. However, by considering the subsequences overlap score (0.4), it is possible to distinguish these two sentences and to assign them to different clusters.

In addition, the word overlap score does not take into account occasional leaps between two sentences, that do not influence the information they convey. Example 3.13 shows that, despite there are very tiny differences between both sentences, this method can be aware of those and identify the two sentences as having similar meanings. These two sentences convey the same information, but their subjects are different, that is, one of the sentences has a definite description (or eventually, for instance, a pronoun), instead of the person's name.

Example 3.13

Sentence#1:	**Sentence#2:**
Tentámos demovê-lo, mas o atleta não se mostrou cooperante num entendimento.	*Tentámos demovê-lo, mas o João nunca se mostrou cooperante num entendimento.*
We tried to dissuade him, but the athlete was not in a cooperative understanding.	We tried to dissuade him, but John never showed a cooperative understanding.

Word Overlap	Subsequences Overlap	Similarity Score
0.69	0.81	**0.75** ≥ 0.75

As they convey similar information, both these sentences should be considered similar. However, by considering only the word overlap score, both sentences would not be considered similar. Thus, another dimension was needed to ensure that, in such cases, sentences are clustered together. That is the role of subsequences overlap score. Taking into account also the subsequences overlap score, it is possible to detect the similarity between both sentences in this Example.

The **subsequences overlap score** (detailed in Equation 3.4) is inspired in *ROUGE-L* (see Section 2.1.5). Firstly, all the subsequences common to both sentences are obtained. Then, the number of words of each subsequence is divided by the number of words of each sentence and all these values are summed up (cf. Equation 3.3). Finally, the sum of all the subsequences is divided by the total number of subsequences found, defining this way the subsequences overlap score.

The same way as the word overlap, the subsequences overlap alone does not reflect the similarity, as two sentences sharing many subsequences might not be considered similar. The subsequences overlap score rewards sentences sharing many subsequences. However, if these sentences do not share the most of their words (given by the word overlap score), they should not be considered similar.

Consider Example 3.14. Despite Sentence#1 is contained in Sentence#2, both sentences are not similar, since Sentence#2 conveys several extra information compared to Sentence#1, thus they must not be assigned to the same cluster.

Example 3.14

Sentence#1:
O director-desportivo do Sporting afirmou hoje que o clube leonino tentou manter João Moutinho no plantel.

The Sporting manager said today that the club tried to keep João Moutinho in the squad.

Sentence#2:
O director-desportivo do Sporting afirmou hoje que o clube leonino tentou manter João Moutinho no plantel, mas o atleta nunca se mostrou cooperante num entendimento.

The Sporting manager said today that the club tried to keep João Moutinho in the squad, but the athlete has never showed a cooperative understanding.

Word Overlap	Subsequences Overlap	Similarity Score
0.64	0.82	**0.73** < 0.75

Finally, the sentences in Example 3.15 (as described above) should be considered similar for the present purposes. These sentences share most of the words, but there are some leaps between them (*hoje*, *RDC*, *nesta sexta-feira*). Both the word and the subsequences overlap scores are high, because these two sentences share many words. Therefore, the similarity score is above the threshold, determining that the sentences are similar.

Example 3.15

Sentence#1:
Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.

A crash in the town of Bukavu, in the eastern Democratic Republic of Congo, killed 17 people on Thursday afternoon, said today a spokesman for the United Nations.

Sentence#2:
Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.

A crash in the town of Bukavu, in the eastern Democratic Republic of Congo (DRC), killed 17 people on Thursday afternoon, said on Friday a spokesman for the United Nations.

Word Overlap	Subsequences Overlap	Similarity Score
0.62	0.87	**0.75** ≥ 0.75

The combination of the word overlap with the subsequences overlap seeks to optimize the creation of the clusters in order to group sentences that convey the same information.

3.3.2 Clustering sentences by keywords

In previous experiments (Silveira and Branco, 2012), human evaluators were confronted with summaries generated automatically by GISTSUMM (cf. Section 2.1.7). When asked about the quality of the summaries, they pointed out that some sentences seemed irrelevant and that they were poorly related to each other. A procedure that effectively selects the most relevant content is very important to improve the quality of a summary.

Many studies sought to find relevant information within large collections of text using combinations of several weighting metrics, as term frequency [(Radev et al., 2000), (Farzindar et al., 2005), (Mori, 2005)], sentence position [(Hovy and Lin, 1999), (Lin and Hovy, 2000), (Mori, 2005)], inverse sentence frequency [(Pardo et al., 2003), (Leite and Rino, 2008)], query overlapping [(Mori, 2005), (Farzindar et al., 2005)], etc. Other studies looked for more sophisticated approaches by using statistical models (LDA, hLDA) to find the relevant topics within a collection of texts (Arora and Ravindran, 2008).

In this work, we have built a simple algorithm using statistical metrics. In order not only to find the relevant information within the collection but also to define a structure in the unrelated data. A clustering procedure based on word frequency is then executed.

SIMBA aims to produce a generic summary, that is a summary that reflects the main idea expressed in the collection of input texts, without focusing in a specific matter. Accordingly, the keywords that represent the collection of texts must be identified. This procedure will be described in the next subsection, before describing the clustering algorithm in the subsequent subsection.

Computing the keywords

Keywords are determined in four steps. The first step adds to a list of candidates the words occurring in the input texts. The words are added to this list considering their lemmas, to ensure that the set of candidates only contains unique words. The second step filters the list of candidates by selecting only the common and proper nouns, since the words in these categories provide good indicators of ideas or themes mentioned in the collection of texts. The third step orders the list of candidates by their word score (*tf-idf*). If two words have the same score, the word frequency is the value considered to untie the word order in the list. In the final step, a predefined number of keywords is retrieved in order to build the final set of keywords, that is used in the keyword clustering. The number of

keywords that is added to the final keywords list depends on the total number of words in the collection of texts. This number is computed using Equation 3.11.

$$k = \sqrt{\frac{N_s}{2}} \tag{3.11}$$

where

N_s – total number of words in the summary.

This is a relevant number since the number of keywords determines the number of clusters that will be created in this clustering step. The number k of keywords is a very studied one in clustering analysis. Many values for k have been proposed. The rule of thumb sets k to the number in the Equation 3.11 (Mardia et al., 1979). Also, $k = 50$ is a very common value since, as stated in (Wives, 2004) and (Schütze and Silverstein, 1997), this value optimizes the clustering algorithm. The number in Equation 3.11 depends on the total number of words that will be included in the summary.

After the set of keywords that represents the collection of texts has been selected, these keywords are rewarded, so as the sentences containing them. The *extra score* of each keyword occurring in each sentence is thus updated. This is a very important step specially when considering that, in the post-processing procedure (cf. Chapter 4), more specifically in the sentence reduction module (cf. Section 4.2.1), parts of the sentences are removed taking into account the sentence *relevance score*. This step aims then to discourage the removal, during the reduction process, of parts of sentences that contain keywords.

In addition, by rewarding the keywords in the sentence, we are also rewarding the sentence itself. The assumption here is that sentences with more keywords tend to be more relevant.

Recall the working example that is being used to describe the summarization process. The keywords obtained for these texts are shown in Table C.5. The sentences obtained in the similarity clustering phase that will be clustered by keywords are shown in Table C.4 (in Annex C), ordered by their *relevance score*. Considering the keywords obtained, and after their *extra score* has been updated, the collection of sentences is ordered differently as detailed in Table C.6. This new order reflects the importance of having a keyword in a sentence, suggesting that sentences containing more keywords are better representative of the key information expressed in the input texts.

Clustering procedure

Once the clustering by similarity has been completed and the keywords have been identified, the representative sentences of the similarity cluster are again clustered, but now based on the set of keywords. A cluster is identified by a keyword (the topic), and contains a representative sentence, and a collection of values (the sentences related to the keyword). Considering that the procedure must take a collection of sentences (retrieved from the previous phase) and a set of keywords that determine the topics of this collection, the algorithm adopted is K-means (MacQueen, 1967), a partitional algorithm that, based on a collection of data, creates a set of clusters whose content is close to each other. The elementary k-means algorithm comprises the following steps:

1. Choose the number of clusters, k;
2. Generate the k clusters centers randomly;
3. Assign each point to the nearest cluster center;
4. Recompute the clusters center;
5. Repeat the two previous steps until a convergence criterion is met.

Our keyword clustering algorithm is an adapted version of K-means. It performs clustering by keywords following the steps described below:

1. Choose the number of clusters, k, defined by the number of keywords;
2. Create the initial empty clusters, represented by each keyword;
3. Consider each sentence:
    a) Compute the occurrences of each keyword in the sentence;
    b) Assign the sentence to the cluster whose keyword occurs more often in it (if there is a tie, the sentence is added to the keyword with the highest score);
4. Recompute the cluster representative sentence. If the current sentence has more occurrences of the keyword defining its cluster than the previous representative sentence had, the newly added sentence becomes the cluster representative sentence; if it has not, the cluster representative remains the same;
5. If the sentence does not contain any keywords, it is added to a specific set of sentences which do not have any keyword ("no-keyword" set);
6. Repeat previous steps (3 – 5) while there are sentences to be processed.

As in the similarity algorithm, when the representative sentence changes, the *extra scores* of both the previous representative sentence and of the new one are updated. The new representative sentence is rewarded, by adding the predefined value to its *extra score*, and the previous representative sentence is penalized, by removing the predefined value from its *extra score*.

In addition, another value is added to the *extra score*: the number of keywords in the sentence. The idea behind this is that the more keywords a sentence has, the more relevant it should be considered. So, the *extra score* of each sentence of each cluster is updated by adding to it the number of keywords occurring in the sentence.

In the same way, a penalty operation is also performed in the sentence *extra score*. Recall from Section 2.1 that Schiffman et al. (2002) proposed a metric that penalizes sentences with less than fifteen words. We also apply such a penalty to sentences with less than fifteen words, while sentences with more than fifteen words are rewarded. Sequences with less than fifteen words are typically considered as conveying less information. In some cases, they are not even full sentences. Titles, subtitles, or headers are examples of such sentences. In addition, this kind of sentences have high scores, since they typically contain very frequent words. Finally, these sentences normally include less information than other sentences containing the same words.

The keyword clustering process is depicted in Figure 3.3.

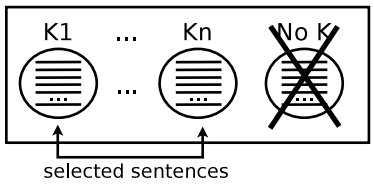


Figure 3.3: Keyword clustering.

Once the algorithm terminates, two types of sentences have been identified: the most significant sentences and the ones to be discarded. The sentences enclosed in a keyword cluster are considered to be the most significant ones of the whole collection. The ones added to the "no-keyword" cluster are ignored.

Recalling our working example, Table C.6 shows the sentences before applying the keyword clustering procedure. Table C.7, in turn, details the clusters obtained after this clustering phase. These are the sentences that will be taken into account in the next steps of the summarization procedure.

The sentences in the cluster NO-KEYWORD are less important to the key information conveyed by the texts to be summarized, as they do not contain any keyword. Hence, they will not be used in the next stages of the procedure.

Note that there are clusters without sentences (BUKAVU and CONGO), which means that these keywords occur less often in the sentences to be clustered, than the other keywords, or their score is lower than the score of the other keywords.

Also, the first sentence in each cluster, its representative, has been rewarded through its *extra score*. Thus, the sentences that have more keywords – for instance, the sentence in the cluster (PORTA-VOZ) – have higher scores.

Finally, the sentences with less than fifteen words appear in the end of the list.

The final list of sentences, the output of this clustering procedure, is illustrated in Table C.8. Note that the sentences in the NO-KEYWORD cluster are not included in this list, as they will not be included in the input for the next steps of the summarization process.

There are several applications of this clustering representation. The sentences grouped in each keyword cluster are related by the key information they commonly address. This relation can thus help to define a topic, and each topic can be represented in the final summary as a paragraph, for instance. Also, the cluster score suggests the importance of its group of sentences within the summary. So, the position of each paragraph in the summary can be defined by its cluster score.

Concluding, this clustering procedure helps not only to select content but also to define the final organization of the summary.

3.4 Summary generation

The ultimate objective of the Summary Generation phase is to create a fluent summary. At this point of the process, two main goals of multi-document summarization have been accomplished: redundancy removal and relevant content selection. Hence, this phase seeks to address the problem of determining which of these relevant sentences can indeed be placed in the summary, meeting a desired compression rate, while building a cohesive text, that fits the users' needs.

Along with retaining the required amount of sentences extracted in the previous phase, a post-processing procedure is applied to the selected sentences in order to create a more readable text. From this point onwards, two desiderata are handled: while text compres-

sion seeks to select which candidate sentences will figure in the summary, post-processing procedure takes the sentences selected and transforms them into a text.

Firstly, Section 3.4.1 describes the text compression step, including its ordering method. Afterwards, Section 3.4.2 introduces the post-processing module, that will be detailed in Chapter 4.

3.4.1 Text compression

Text compression includes two stages executed in sequence: sentence ordering and sentence selection.

The order of the sentences, extracted from different texts and from different points in those texts, in the final summary is a crucial step to assure that the text is not hard to read or even to make sense of. Sentence ordering is a complex task to accomplish automatically, specially when considering that the sentences are retrieved from different sources. Thus, deciding whether a sentence selected from a document should be placed before or after another sentence retrieved from a different document is not a straightforward task.

The original order – the order of the sentences in the source texts – cannot be used, since sentences come from different documents and their positions are not comparable. Yet, Okazaki et al. (2004) propose a method to order sentences of a multi-document summary based on the publication date of the source texts. In a preliminary phase, chronological order is used. Then, an ordering refinement is done based in a precedence relation procedure. This requires a previous annotation of the source texts regarding the publication date. However, in general, no such annotation is available.

In texts from news, for instance, the most relevant sentences appear first in the text. The approach pursued in this work follows this approach. Considering that users are expecting to get firstly the most important information, the sentence score was used to define the final order of the sentences in the text.

Recall from previous sections that the sentence score has two dimensions: the *main score* and the *extra score*. While the *main score*, as it depends on the *tf-idf* scores of the sentence words, measures the relevance of the sentence considering the whole collection, the *extra score* reflects the decisions made during the summarization process. Both these dimensions are relevance metrics, so that their combination is used to determine which set of sentences will be selected to figure in the summary. Hence, all the sentences

obtained in the content selection phase are ordered considering their *relevance score*.

Taking an ordered collection of sentences, the compression procedure selects from it the number of sentences that fulfills the compression rate required by the user. In case the user does not submit the compression rate value, in the literature, the default value is set to 70%, which is the commonly used default value since it ensures that the summary may contain the minimal relevant information within the text collection.

As stated in Section 2.1, in our work, a low compression rate is a value that maintains in the summary most content of the input texts, that is a compression of 1% is a low one. Otherwise, a high compression rate, for instance 99%, reduces the original text by 99%, thus creating the summary with only 1% of the original texts.

In order to meet the compression rate, we consider words as the smallest units of the text. Other works, such as GISTSUMM (Pardo et al., 2003), for instance, have used the sentence as minimal unit. Depending on the system goals, both options can be adopted. Taking into account words tends to be a fairer metric, in the literature.

Thus, considering the total number of words contained in all the texts, and taking the compression rate submitted by the user (or the default value), the number of words that should be in the summary is computed, using Equation 3.12.

$$summaryWords = compressionRate \times totalWords \tag{3.12}$$

where

summaryWords	–	number of words in the summary.
compressionRate	–	compression rate.
totalWords	–	number of words in the collection of texts.

Afterwards, sentences are added to the summary by taking into account the number of words they contain, until the total number of words for the summary has been met.

A final decision has to be made concerning the last sentence to be added to the summary. While the total number of words of this sentence exceeds the total number of words of the summary, two options can be taken: (1) either the sentence can be cut from the word that attains the compression rate until the end of the sentence; (2) or the sentence can be added without being modified and the compression rate will not be met. Both decisions have implications. If option (1) is selected, the summary might not be as cohesive and fluent as possible, though the desired compression rate can be obtained. Otherwise,

if (2) is the option considered, there are more chances that a cohesive and fluent text can be ensured, but the summary will not contain exactly the number of words defined by the compression rate.

The main goal of our work is to produce a summary to be consumed by a human reader. If a sentence would be cut in two parts, this goal could not be achieved. So, we accept that the exact compression rate be compromised, by adding the complete last sentence to the summary, so that a better summary can be produced for humans.

Moreover, while the sentence reduction process (cf. Section 4.2.1) removes words from the sentences, the connective insertion procedure (cf. Section 4.3.2) will add words to the final set of sentences. Due to all this, the post-processing step can also affect the achievement of the compression rate.

Measuring the pros and cons of these options, the gains associated with having a better text would prevail over retrieving the exact compression rate.

Recall the input texts shown in Examples 3.1 and 3.2 from Section 3.1. After executing the content selection phases (cf. Section 3.3), the sentences are ordered as shown in Table C.8. When applying a default compression rate (70%), the sentences in Table C.9 define the summary returned by SIMBA. Without further post-processing, Example 3.16 illustrates the final summary.

Example 3.16

Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.
O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.
Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.

A compression rate of 70% states that, considering the number of words in the input texts (252), the summary must have at most 76 words. However, this summary contains 82 words, 6 words more than desired. If the last sentence would not have been added to the summary we would not comply with the compression rate, as the summary would contain 55 words. If we had truncated the last sentence in the last six words to meet exactly the compression rate, we would include a sequence of words that would not make sense. By measuring the pros and cons, to include the complete sentence in the summary has been considered the best option, despite slightly compromising the compression rate.

3.4.2 Post-processing

As it was stated in the previous section, at this point in the processing, there is already a summary to be delivered. However, it not only includes more words than the desired compression rate, but also its textual quality can be improved.

The post-processing procedure is composed by two types of tasks: compression tasks – sentence reduction and compression review –, and fluency tasks – paragraph creation and connective insertion.

Sentence reduction seeks to remove possibly nonessential information within each sentence in the summary. Compression review aims to complete the summary with novel sentences after sentences already in the summary have been reduced.

Considering the already reduced sentences, paragraph creation groups these sentences in paragraphs defined around topics. Finally, connective insertion creates a relation between adjacent sentences within paragraphs.

These post-processing module is described in detail in the next Chapter 4.

3.5 Summary

This Chapter detailed SIMBA, an automatic multi-document summarization system for the Portuguese language. SIMBA takes a collection of texts as input and delivers a summary that expresses the key information conveyed by those texts.

Multi-document summarization is performed by executing two phases in sequence: the first phase, described in the current Chapter, determines the sentences to be included in the summary, while the second phase, described in the next Chapter, seeks to improve the textual quality of the final summary delivered to the user.

In the first phase, two clustering procedures are performed aiming at fulfilling the two main challenges of multi-document summarization: removal of redundancy and selection of relevant information.

The first clustering procedure, the similarity clustering, identifies redundant information, while the second clustering procedure, the keyword clustering, selects the most relevant content to be included in the summary.

Similarity clustering starts by collecting all the sentences from the input texts and compares them based on a similarity measure, in order to determine if the sentences

convey the same information. If they do, they are considered similar, and are grouped in the same cluster, from which only one sentence is selected.

Afterwards, the keyword clustering procedure takes the redundancy free sentences previously selected and groups them in relevant topics, based on the keywords obtained in the collection of texts. This way, the relevant information is identified and organized in clusters of topics.

After this is finished, a summary can be delivered to the user. However, this summary can still be improved by removing redundancy within sentences, by grouping sentences in topics, and by creating relations between the sentences. This is accomplished by the second phase, the post-processing one, that is presented in the next Chapter.

CHAPTER 4

POST-PROCESSING

Our work aims at improving the quality of summaries produced by extraction methods. The textual quality of the final summary (e.g. fluency, readability, cohesion, etc.) has been repeatedly reported as the main flaw in current automatic summarization technology.

Several anomalies were identified when manually evaluating automatic summaries (Silveira and Branco, 2012). Human judges noted that the summaries presented to be evaluated contained very long sentences and that the sentences were not very well interconnected, which compromised the fluency of the summary.

The major goal of this Chapter is to address these problems and seek for improvements in this respect. The issue of which sentences to extract for the summary (cf. Chapter 3) has been extensively researched in the literature in the last decades. The increasingly more sophisticated algorithms for extraction seem to bring now little improvements, if any, as it is definitely not in their nature to cope with the problem of textual quality. In our view, the quality of the texts extracted is where there is room for improvement.

Accordingly, we developed a method to enhance extraction-based summaries by means of two types of tasks (compression tasks and fluency tasks), that together seek to improve the textual quality of the final summary. The rationale behind this post-processing procedure is then to combine several solutions that together have the promise to allow for progress in this path of research.

Compression tasks seek to enhance what could be called the information density of the summary. These tasks are described in Section 4.2 and include sentence reduction (cf. Section 4.2.1) and compression review (cf. Section 4.2.2). **Fluency tasks**, which aim to improve the textual quality of the text, are described in Section 4.3 and include paragraph creation (cf. Section 4.3.1) and connective insertion (cf. Section 4.3.2).

Firstly, Section 4.1 describes the process that joins together these tasks. Afterwards, the procedures composing these tasks are described in detail in Sections 4.2 and 4.3. Finally, Section 4.4 discusses the decisions and methods used.

4.1 Overview

The **post-processing** phase combines two types of procedures: compression tasks and fluency tasks that are executed in this sequence.

Post-processing starts after the sentences for the summary have been selected. All these sentences will first be reduced in their length, while still seeking to preserve and convey their main content (sentence reduction). As this step removes words from sentences, the summary compression rate will likely be compromised, that is the summary could have less words than the compression rate setup determines. Therefore, after the first set of sentences have been reduced, a new set of sentences must be added to the summary until the number of words defined by the compression rate has been met again (compression review). This new set of sentences will afterwards be reduced. The compression tasks have to be repeated until the compression rate has been finally reached.

Afterwards, fluency tasks are executed. Firstly, the sentences are grouped into paragraphs by means of the insertion of paragraph breaks. Then, in each paragraph, discourse connectives may be inserted between sentences sharing a discourse relation.

These post-processing steps are described in detail in the following sections.

4.2 Compression tasks

Compression tasks define two procedures seeking to include in the summary as much information as possible.

Firstly, shorter sentences are built out of the extracted sentences, by means of a sen-

tence reduction procedure (described in Section 4.2.1). Afterwards, the compression review procedure (described in Section 4.2.2) makes sure that the summary fulfills the requested compression rate.

The following subsections go through both these procedures in detail.

4.2.1 Sentence reduction

As was discussed in the previous Chapter 3, the most common solution for complying with the compression rate is to allow the last sentence to be truncated, where the exact compression rate has been reached in terms of number of words, compromising the fluency and grammaticality of the summary, and thus the quality of the final text. An alternative is the one where the last candidate sentence is kept in full, surpassing the compression rate. None of these solutions is optimal.

We propose to use sentence reduction to compress the extracted sentences down to their key content. This allows for more sentences to enter in the summary, thus producing a more comprehensive text. After the summarization process has identified and ranked the most relevant sentences, linguistic structures in these sentences, that tend to convey information less essential to figure in the limited space of the summary, may be removed.

The rationale behind using sentence reduction in a summarization process is thus twofold. On the one hand, it removes expendable information, generating a simpler and easier to read text with a likely higher key information density. At the same time, this allows for the addition of more individual (reduced) sentences to the summary, that otherwise would have not been included.

Consider the ranked list of sentences in Example 4.1, which are candidates extracted to be added to a summary.

Example 4.1

1. EU leaders signed a new treaty to control budgets on Friday.
2. Only Britain and the Czech Republic opted out of the pact, signed in Brussels at a summit of EU leaders.
3. UK Prime Minister David Cameron, who with the Czechs refused to sign it, said his proposals for cutting red tape and promoting business had been ignored.
4. The countries signed up to a promise to anchor in their constitutions – if possible – rules to stop their public deficits and debt spiraling out of control in the way that led to the eurozone crisis.
5. The treaty must now be ratified by the parliaments of the signatory countries.

Considering a given compression rate, only the four first sentences would be included in the final summary. However, there are particular stretches that can be removed from these sentences making room for the inclusion of more relevant sentences, with more information. Appositions, parenthetical phrases and relative clauses are examples of those stretches. Consider, for instance, the following expressions candidates for removal:

- The parenthetical phrase: *signed in Brussels at a summit of EU leaders*
- The relative clause: *who with the Czechs refused to sign*
- The parenthetical phrase: *if possible*

If these expressions are removed from the respective sentences, there will be room to include yet another sentence. Otherwise that fifth sentence would not be included in the final summary, despite bringing relevant information not conveyed yet by the other previous four sentences.

A new version of the extracted summary, in which sentences have been reduced, is shown in Example 4.2.

Example 4.2

EU leaders signed a new treaty to control budgets on Friday.
Only Britain and the Czech Republic opted out of the pact.
UK Prime Minister David Cameron said his proposals for cutting red tape and promoting business had been ignored.
The countries signed up to a promise to anchor in their constitutions rules to stop their public deficits and debt spiraling out of control in the way that led to the eurozone crisis.
The treaty must now be ratified by the signatory countries' parliaments.

As exemplified with this resulting summary, it is possible to produce a summary containing more key information conveyed by the original collection of texts.

Evaluation experiments undertaken by human users (Silveira and Branco, 2012) have shown that sentence reduction indeed helps to improve the summaries produced. Source texts were randomly selected from *TeMário* (Rino and Pardo, 2003), a corpus of Portuguese texts containing a summary for each text. Automatic summaries were build using GISTSUMM (cf. Section 2.1.7). Then, reduced summaries were built by hand, from the automatic summaries by removing parenthetical and apposition phrases from their sentences. This procedure was manually performed and aimed at simulating the process an automatic sentence reduction tool would execute. Human judges were then asked to rate

the reduced summaries. In a first task, they compared them with the source texts, in order to assess the quality of the reduced summaries considering the source texts. In a second task, reduced summaries were compared with the original automatic summaries (before being manually reduced), in order to assess if the reduced summary, though with shorter sentences, still conveys the key information as the automatic one. Results have shown that reduced summaries could replace the original automatic summaries, since they preserve not only the idea of the original text and the content of the original automatic summary, but also they allow the inclusion of more key information in the summary.

The same way, in a pilot study (Lin, 2003), Lin showed the potential of sentence reduction to improve a multi-document summarization system, using a noisy-channel model approach. Also, Berg-Kirkpatrick et al. (2011) used a machine learning approach to perform sentence extraction and compression for multi-document summarization, which proved to be effective in improving the quality of the summaries produced.

In a different perspective, Marsi et al. (2010) argued that "a hybrid approach to sentence compression – explicitly modeling linguistic knowledge – rather than a fully data-driven approach" is the better way to perform sentence reduction.

In our approach presented here, we adopt a hybrid approach by combining a statistical parser, LX-Parser (cf. *Annex B.2*), with rules to be applied on the output of the parser that target the linguistic structures to be taken into account in this procedure.

Sentence reduction condenses thus the initial summary in order to produce a new text containing simpler and more concise sentences.

In order to perform reduction, several types of syntactic structures are considered. The identification of these structures is described in Section 4.2.1.1, and the algorithms experimented with are detailed in Section 4.2.1.2.

4.2.1.1 Linguistic structure identification

There are a number of structures that can be seen as containing "elaborative" information about the content already expressed. In this work, four types of structures are targeted:

- Parenthetical phrases;
- Appositions;
- Sentence-level prepositional phrases;
- Appositive relative clauses.

These structures are targeted taking into account the parse tree of each sentence, which is delivered by the syntactic parser. These structures are identified in the tree using *Tregex* (Levy and Andrew, 2006), a utility for matching patterns in trees. *Tregex* takes a parse tree and a regular expression pattern, and returns the subtrees of the input tree that are matched by the input pattern.

Parenthetical phrases: These are phrases that explain or detail other information being expressed. The sentence in Example 4.3 contains a parenthetical phrase.

Example 4.3

O Parlamento aprovou, por ampla maioria, a proposta.
The Parliament approved by large majority the proposal.

Parenthetical phrases can be introduced by a preposition (`PP`), or an adverb (`ADV` or `ADVP`), or a conjunction (`CONJ` or `CONJP`) (cf. the parse tree for this sentence in *Annex D.1.1*). These phrases are enclosed by punctuation symbols, as parenthesis, commas or dashes.

Appositions: These can be seen as a specific type of parenthetical expressions, composed by noun phrases that describe, detail or modify their antecedent (also noun phrases). The sentence in Example 4.4 contains an apposition.

Example 4.4

José Sócrates, primeiro-ministro, e Jaime Gama querem cortar os salários dos seus gabinetes.
José Sócrates, the Prime Minister, and Jaime Gama want to cut the salaries of their offices.

Appositions are identified by noun phrases (`NP`) or adjective phrases (`AP`) typically enclosed by two commas or two dashes (cf. the parse tree for this sentence in *Annex D.1.2*).

Sentence-level prepositional phrases: These are phrases headed by a preposition used to include additional information in sentences. The sentence in Example 4.5 contains a prepositional phrase of this kind.

Example 4.5

No Médio Oriente, apenas Israel saudou a operação.
In the Middle East, only Israel welcomed the operation.

Prepositional phrases (PP) we will be targeting are introduced by a preposition and are followed by a sentence (S) that contains the subject and the main verb (cf. the parse tree for this sentence in *Annex D.1.3*).

Appositive relative clauses: These are clauses that detail the information conveyed by a noun phrase. The sentence in Example 4.6 contains a relative clause.

Example 4.6

O Parlamento aprovou a proposta, que reduz os vencimentos dos deputados.
The Parliament approved the proposal, which reduces the salaries of deputies.

Appositive relative clauses are introduced by a relative pronoun and their structure is defined by a complementizer phrase (CP) that has to be preceded by a comma (cf. the parse tree for this sentence in *Annex D.1.4*).

4.2.1.2 Algorithms

Two types of algorithms for sentence reduction were tested: BLIND REMOVAL and SCORE-BASED REMOVAL. Both these algorithms take the original sentence and their previously identified structures, and return a reduced sentence.

After performing reduction, both the original sentence score and the reduced sentence score are compared to check if the reduced sentence is better than the original one. If it is, the reduced version of the sentence is the one chosen. If not, the original sentence is kept and the reduced version is not used.

Blind removal is an algorithm that takes all the structures detailed in Section 4.2.1.1 and removes[1] them from the original sentence.

Consider the sentence in Example 4.7, whose removable passages are underlined.

Example 4.7

Também hoje, na conferência de líderes, o ministro dos Assuntos Parlamentares, Jorge Lacão, afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora.
Today also, at the leadership conference, the Minister for Parliamentary Affairs, Jorge Lacão, said to have discovered that the office of the prime minister had been excluded.

[1] In this context, "remove" means that a given subtree in a parse tree is replaced by a null tree.

With this algorithm, all these passages are removed from the original sentence, building the reduced sentence illustrated in Example 4.8.

Example 4.8

Também hoje, o ministro dos Assuntos Parlamentares afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora.
Today also, the Minister for Parliamentary Affairs said to have discovered that the office of the prime minister had been excluded.

Score-based removal is an algorithm that approximates the notion of power set. In mathematics, a power set is the set of all subsets of a given set. Here, the "reduced power set" of a given sentence is used to refer to the set with all the reduced sentences that is possible to obtain by combining the removal of its individual candidate structures.

Example 4.9 shows the sentence from Example 4.7 and its score. Example 4.10 describes its power set and the respective scores of each member sentence.

Example 4.9

Também hoje, na conferência de líderes, o ministro dos Assuntos Parlamentares, Jorge Lacão, afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora.	1.7200

Example 4.10

Também hoje, na conferência de líderes, o ministro dos Assuntos Parlamentares afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora. Today also, at the leadership conference, the Minister for Parliamentary Affairs said to have discovered that the office of the prime minister had been excluded.	1.8175 (AP removed)
Também hoje, na conferência de líderes, o ministro dos Assuntos Parlamentares, Jorge Lacão, afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora. Today also, at the leadership conference, the Minister for Parliamentary Affairs, Jorge Lacão, said to have discovered that the office of the prime minister had been excluded.	1.7200 (original sentence)
Também hoje o ministro dos Assuntos Parlamentares afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora. Today also the Minister for Parliamentary Affairs said to have discovered that the office of the prime minister had been excluded.	1.7053 (AP and PP removed)
Também hoje o ministro dos Assuntos Parlamentares, Jorge Lacão, afirmou ter-se descoberto que o gabinete do primeiro-ministro tinha ficado de fora. Today also the Minister for Parliamentary Affairs, Jorge Lacão, said to have discovered that the office of the prime minister had been excluded.	1.6000 (PP removed)

After the "reduced power set" has been determined, the different sentences are ordered by their *relevance score*. As illustrated in Example 4.10, depending on the passage that has been removed or the combination of passages removed, the score of the reduced sentence is different. This means that there are some expressions that may be seen as containing more key information than others, as the *relevance score* is an indication of informativeness. Note that, for instance, the original sentence, from which were built the reduced ones, is in the second place in the list. Note also that the third reduced sentence in the list was obtained by removing all the possible targeted passages, being the same sentence that the BLIND REMOVAL algorithm would create (cf. Example 4.8). Its *relevance score* is lower than either the best *relevance score* of a sentence in the list or the *relevance score* of original sentence.

The reduced sentence to be kept will then be the sentence in the "reduced power set" that has the maximum *relevance score*.

Discussion

The main assumption of a reduction process is that the identified syntactic structures are candidates to be removed because they typically detail accessory information with respect to the key information conveyed by that sentence.

Taking this into account, the two algorithms presented before were tested. BLIND REMOVAL considers that all the information targeted can be dispensable. Thus, all the candidate stretches are "blindly" removed from the sentences that go through this process.

Considering the nature of these stretches, their removal could make room for more information to be included in the summary. However, under closer scrutiny after applying this algorithm, there may be some stretches that, if removed, would compromise the comprehensiveness of the text. Too much information may be being removed, even information that is relevant to be in the summary. This way, the reduction procedure would not have the impact required, that is to identify and remove dispensable information.

SCORE-BASED REMOVAL, in turn, aims to avoid this pitfall. By removing the structures taking into account the impact of that removal on the score of the sentence, it is expected that the informativeness of the summary also improves. The *relevance score*, by being an indicator of the sentence informativeness, determines which of the reduced sentences created is the best, that is, the one that contains higher key information density.

SCORE-BASED REMOVAL was then the algorithm selected, since it verifies two important conditions: (1) produces the best combination of truncated stretches to obtain a reduced sentence, and (2) it takes into account the *relevance score* of the resulting reduced sentences.

Nevertheless, by being a naïve approach to sentence reduction, BLIND REMOVAL will be used in the evaluation procedure (cf. Chapter 5) as a baseline to test the efficiency of the SCORE-BASED REMOVAL algorithm.

Recall the texts in Examples 3.1 and 3.2 from Section 3.1. The sentences in Table C.8 (cf. Annex C) define a summary that could be delivered to the end-user. However, its textual quality can be improved, and the first step towards this goal resides in the current sentence reduction procedure. The sentences illustrated in Table C.12 have already been reduced. Take for instance sentence#2. It contains an apposition phrase (*Trasept Congo*), containing a keyword (*Congo*) (cf. Table C.5). This apposition could have been removed. However, when computing the *relevance score* of the candidate reduced sentence, it would have a lower *relevance score* than the original sentence, and that is why the original sentence has been kept (cf. Table C.10). The inverse has occurred with sentence#3. The original sentence contained a parenthetical phrase (*prejudicado pelo mau tempo*). The *relevance score* of the reduced sentence, produced by removing this phrase, is higher than the one of the original sentence. Thus, the original sentence was replaced by this reduced sentence (sentence#3) (cf. Table C.11).

It is worth a final look at the differences between Tables C.9 and C.12. Not only sentence#3 is a newly added sentence to the summary, but also it has been reduced. In fact, by removing dispensable stretches from the sentences, it is possible to include more information in the summary, through the compression review step.

4.2.2 Compression review

As stated in Section 4.1, the list of sentences that has been reduced composes the summary. Nevertheless, at this point, the summary contains less words than the number of words defined by the compression rate. This makes room for novel information to be added to the summary.

In Section 3.4.1, a compression procedure is described. A list of ranked sentences was split into two sublists at the point where the number of words that the compression rate

for the summary defines. The first sublist contained the sentences in the summary to be post-processed. The second sublist contained a set of reserved sentences that can be added to the summary if needed. The sentence reduction procedure took the first sublist and reduced its sentences to their main content.

The compression review step takes the second reserved sublist and adds new sentences to the summary until the compression rate will be complied with again. Now, these new sentences also need to be reduced. So, compression review defines a cycle where sentences are firstly added to the summary, and then are reduced.

Summing up, both these steps, that add new sentences and reduce them, are repeated until no more (reduced) sentences can be added to the summary.

4.3 Fluency tasks

Fluency tasks seek to improve the cohesion and fluency of the text. On the one hand, the insertion of paragraph breaks (Section 4.3.1) groups sentences that are related to each other in terms of content into paragraphs. On the other hand, the insertion of connectives (Section 4.3.2) creates textual connections between the sentences by means of the insertion of conjunctions or other discursive connectives.

The following subsections go through both these procedures in detail.

4.3.1 Paragraph creation

When taking into account a single source text, there may be many solutions that improve the fluency of its summary. Maintaining the order of the extracted sentences as they occur in the source text to be summarized is the most straightforward approach. However, when considering a set of sentences retrieved from different source texts, some relation must be found between those sentences to create a fluent text, in order to enhance its textual quality.

Recall, for instance the two example texts – Example 4.11 and 4.12 – from Chapter 3 –, a translated version of these texts can be found in *Annex C.1*. Both these texts can be different summaries for the same source texts. Despite there can be similar ways to select the content of the text, the way it is organized also defines the relevance of each element in it.

Example 4.11: Text #1

[1] Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.

[2] As vítimas do acidente foram 14 passageiros e três membros da tripulação. Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu. Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.

[3] O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala. "Não houve sobreviventes", disse Okala. O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.

Example 4.12: Text #2

[1] Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas. As vítimas do acidente foram 14 passageiros e três membros da tripulação. Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 Km do aeroporto de Bukavu.

[2] O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala. "Não houve sobreviventes", disse Okala.

[3] O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.

[4] Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.

These two texts are organized in different ways, highlighting different points of view in different points. For instance, in *Text#1*, paragraph#1 contains only a single sentence, in order to highlight its relevance. Yet, in *Text#2* the same sentence together with other sentences that convey some more information about the event, forms a paragraph (#1). Note also that paragraph#3 in *Text#1* relates all the information obtained in an interview. Finally, it is worth mentioning paragraph#4 in *Text#2*, which is defined by a single sentence and positioned alone in the end of the text, stating that the information it contains is additional regarding the rest of the text. Different ways of organizing the text can have an impact not only on its understanding but also on the key information conveyed.

In a text, paragraphs are discourse units that group sentences related to each other. **Paragraph creation** aims thus to contribute to the enhancement of the textual quality of a summary. At a given point (cf. Section 3.4.1), the ranking of the sentences has already been defined: the sentences considered the most relevant ones appear first in the summary. Finding a way to groups them into paragraphs is then the next step.

Our approach to paragraph creation relies on the organization provided by the keyword clustering procedure. Each sentence that has been selected to be part of the summary was previously associated with a keyword cluster. Each keyword cluster can be seen as representing a topic in the collection of documents, with the sentences that are in each cluster characterizing its respective topic. The rationale behind our approach to paragraph creation is that sentences that are in the same keyword cluster are ready to be grouped into a paragraph.

Hence, for each sentence that composes the summary, the keyword that identifies its cluster is obtained. The order of the sentences in the paragraphs is defined by the ranking of the sentences, so that the highest scored sentence is the first to occur in its paragraph. Then, all the sentences that pertain to that cluster are added. The same procedure is applied to every paragraph. The sentences occurring in each paragraph are, thus, ranked considering their *relevance score*.

In line with what was stated in Section 3.3.2 when discussing the applications to the clustering representation, the ranking of the paragraphs is, in turn, defined by their scores. The score of a paragraph is defined in Equation 4.1

$$score_p = \frac{\sum_{s \in p} relevanceScore_s}{totalSentences_p} \tag{4.1}$$

where

p	–	paragraph.
s	–	sentence.
$score_p$	–	score of the paragraph.
$relevanceScore_s$	–	relevance score of the sentence s.
$totalSentences_p$	–	number of sentences in the paragraph p.

Paragraphs are added to the final summary according to their decreasing *relevance score*, as already discussed.

Recalling our example, Table C.12 shows the sentences after the reduction procedure has been applied. These sentences will now be grouped into paragraphs based on the cluster of keywords to which they were assigned. Table C.13 (in *Annex C*) shows the sentences in their final order and the defined paragraphs (keywords are shown just for reference of the cluster from which the sentences were obtained). Note that the highest scored paragraphs appear first and the sentences within each paragraph are ordered based on their decreasing *relevance score*, as well.

Example 4.13 shows the final summary with the new paragraphs represented.

Example 4.13

Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.

O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, também levava uma carga de minerais. Todos morreram quando o avião, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.

Note that by creating paragraphs it is possible to define topics containing related information, a process that, in conjunction with all the post-processing stages, seeks to enhance the textual quality of the summaries. Further discussion on the decisions made about post-processing stages can be found below in Section 4.4.

4.3.2 Connective insertion

An important research issue where there remains much room for improvement in automatic text summarization is text cohesion. In order to be able to be read with naturalness by a human reader, a text must form a cohesive whole.

Text cohesion is very hard to ensure specially when creating summaries from multiple sources, as their content can be retrieved from many different documents.

The content retrieved from multiple sources must thus be organized in such a way to enhance the cohesiveness of the output summary. However, strategies to improve cohesion in a text have not been addressed in the literature. Most of the work done so far sought mainly to identify and classify discourse relations, in order to improve parsing performance, or content selection in single-document summarization (cf. Section 2.3).

The post-processing step addressed in this section, **connective insertion**, aims to enhance the textual cohesion of a summary produced by an extractive summarization system.

Connectives are textual devices that ensure text cohesion as they support the text sequence by signaling different types of connections or discourse relations among sentences. Examples of discourse connectives can be *mas* (but), *ou seja* (that is), *assim* (hence), *a menos que* (unless), etc. It is possible to understand a text that does not contain any connective, but the occurrence of such elements reduces the cost of processing the information for human readers, as they explicitly mark the discourse relation between sentences, thus acting like guides in the interpretative process of the text.

The assumption here is that relating sentences that are retrieved from different source texts can produce a more interconnected text, and thus a more easier to read summary. Take the sentences in Example4.14.

Example 4.14

S_1 *A CGD e o BPI têm solidez financeira para suportar um cenário adverso para a banca.*
CGD and BPI have the financial strength to withstand an adverse banking scenario.
S_2 *O BCP tem rácios inferiores ao exigido..*
BCP has ratios lower than required.

When connecting these sentences with a discourse connective that expresses an opposition relation (cf. Example4.15) these can produce a more comprehensive and easier to read sequence.

Example 4.15

S_1 *A CGD e o BPI têm solidez financeira para suportar um cenário adverso para a banca.*
CGD and BPI have the financial strength to withstand an adverse banking scenario.
S_2 ***Pelo contrário**, o BCP tem rácios inferiores ao exigido..*
On the contrary, BCP ratios has lower than required.

Pursuing this goal through an automatic procedure is a highly non trivial research task to accomplish. This is even more true when considering a text whose sentences were retrieved from different source texts.

Connective insertion takes a sequence of sentences and possibly inserts between those sentences discourse connectives that explicitly convey discourse relations among them.

So, considering two adjacent sentences, the goal is to insert, between those sentences, some discourse connective that stands for the discourse relation found between them – including possibly the phonetically null one.

Marcu and Echihabi (2002) noted that "discourse relation classifiers trained on examples that are automatically extracted from massive amounts of text can be used to distinguish between [discourse] relations with accuracies as high as 93%, even when the relations are not explicitly marked by cue phrases". Our approach here will be based on a classifier that predicts the relation of two previously unseen sentences. To this end, a corpus of discourse relations will be needed.

The remainder of this Section is organized as follows: Section 4.3.2.1 overviews the connective insertion process; then, Section 4.3.2.3 details the creation of the corpus that supports the training of the classifier for the inserting procedure. Afterwards, in Section 4.3.2.5, the definition of the classifier used and its features is discussed. Finally, Section 4.3.2.6, reports the inclusion of this classifier in the summarization procedure.

4.3.2.1 Overview

In the literature on discourse analysis, most common approaches to identify discourse relations among sentences use machine learning techniques over annotated data. The task is to learn how and which discourse relations are explicitly – by means of cue phrases – or implicitly expressed on human annotated data.

In our work, this task is reverted. The classification of the discourse relation at stake can be used to determine a discourse connective to be inserted between a given pair of adjacent sentences. Consider the sentences in Example 4.16.

Example 4.16

S_1 *O custo de vida no Funchal é superior ao de Lisboa.*
The cost of living in Funchal is higher than in Lisbon.
S_2 ***No entanto**, o Governo Regional nega essa conclusão.*
However, the Regional Government denies this conclusion.

These two sentences are related by the discourse connective *no entanto* ("however"), which expresses that the two sentences convey some adversative information. Hence, it is possible to say that these sentences entertain a certain discourse relation on the basis of the discourse connective that relates them.

To feed the machine learning procedure, an annotated corpus is needed to train a classifier that learns which type of discourse relation holds between two sentences and thus which discourse connective can be inserted between them.

Figure 4.1 resumes the steps leading to the accomplishment of this task, and that will be described in detail in the following sections.

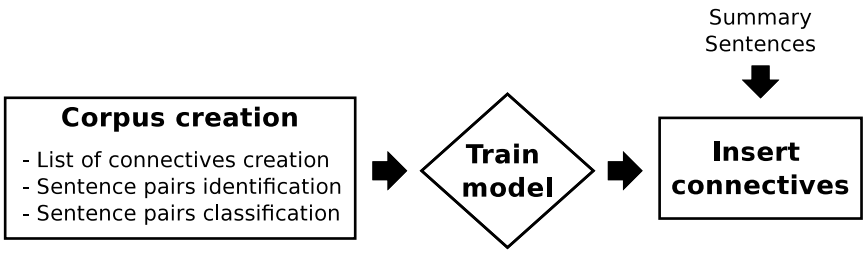


Figure 4.1: Overview.

Considering the enhancement of cohesion in extraction-based summaries, connective insertion is a task that automatically inserts a discourse connective between two extracted sentences (cf. Section 4.3.2.6). In order to insert a connective between two sentences, first we need to identify which sort of discourse relation they entertain. The most common method pursued in the literature to find discourse relations in text has been, as was mentioned above, machine learning.

Our work follows this same path. A classifier was trained (cf. Section 4.3.2.5) with information retrieved from a corpus that explicitly associates two sentences through a discourse relation. However, such corpus is a resource that was not available for the Portuguese language. Thus, we built a discourse corpus composed by pairs of sentences linked by a discourse connective, extracted on the basis of a set of heuristics (cf. Section 4.3.2.3) and run over a corpus of raw texts. As the semantic type of a discourse connective defines the discourse relation that two sentences entertain, the next step associated the relevant type to each pair of sentences in the corpus. The type is one in a list of discourse connectives and their types that was collected by hand (cf. Section 4.3.2.2).

Summing up, a corpus of pairs of sentences and their respective discourse relation was built to feed a classifier that aims to identify a discourse relation shared among two previously unseen input sentences. Considering this relation, a discourse connective is afterwards inserted between those two sentences.

This is a pioneer approach seeking to enhance the textual quality of a summary.

4.3.2.2 List of connectives

A discourse connective may be an element from different grammatical categories (prepositions, conjunctions, adverbs, prepositional phrases, conjunctional phrases, adverbial phrases) that expresses a discourse relationship between sentences.

To the best of our knowledge, no research was undertaken so far concerning the automatic identification of discourse relations in Portuguese. As no previous list was available that classified discourse connectives according to the discourse relation they express, we built a list of Portuguese discourse connectives, inspired on the English *Penn Discourse TreeBank (PDTB)* [(Prasad et al., 2007) and (Prasad et al., 2008)] classification[2].

The *PDTB* contains four main classes of discourse relations expressed by discourse connectives: TEMPORAL, CONTINGENCY, COMPARISON, and EXPANSION. These classes divided in types and subtypes. From this point on, when referring to a discourse relation conveyed by a connective the notation CLASS-TYPE-SUBTYPE will be used. Take for instance the sentences in Example 4.16. These two sentences entertain a discourse relation of COMPARISON-CONTRAST-OPPOSITION.

Firstly, a structured list of connectives was built by translating all the structure list of the *PDTB* and its connectives from English. Afterwards, an inspection was made to our corpus to assess whether the connectives in this list were correctly classified. Considering this review and taking into account the purpose of this task, some adjustments were made to the translated list. Changes included the inclusion of more connectives in each class[3] and the adaptation of some types and subtypes in the original structure. The resulting list of connectives is detailed in *Annex D.2.1*. This process of translation and extension of the *PDTB* list introduced some ambiguity issues that are discussed below in Section 4.3.2.4.

The discourse connective list intends to be as exhaustive as possible. However, at this point, some cases of correlative conjunctions that can be seen as paired connectives, like *não só ... mas também* ("not only... but also"), were not considered, as their structure requires a more elaborate processing of multi-word expressions. Moreover, these paired connectives do not occur very often in the corpus.

Summing up, the resulting list contains a total of 136 discourse connectives. This list seeks to reflect the most common phenomena that express different discourse relations.

[2]This classification was supported in Portuguese by Mateus et al. (2003) and Silvano (2010).

[3]Connectives were obtained in the dictionary and assigned to their corresponding class with the help of an expert in Linguistics.

4.3.2.3 Discourse corpus creation

In this subsection, we describe the creation of a corpus of pairs of sentences and their respective discourse relations.

Figure 4.2 summarizes the process of creation of this corpus.

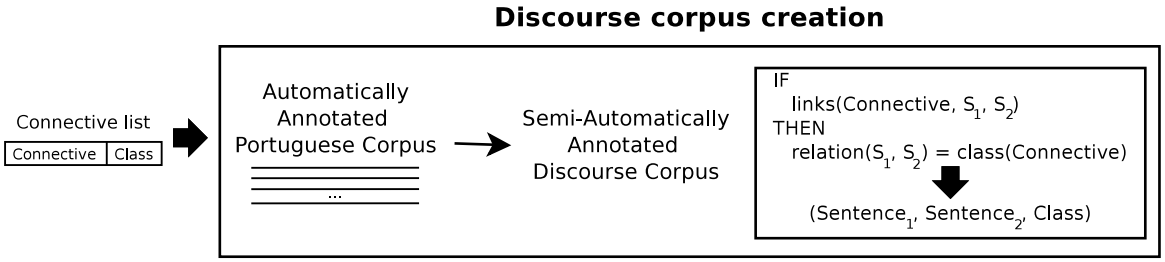


Figure 4.2: Discourse corpus creation overview.

The discourse relation of each pair of sentences is determined by the class of the discourse connective defined in the list of connectives. This way, a corpus containing triples, defined by the two sentences and their respective discourse relation, was created.

Note that we have also accounted for the occurrence of the phonetically null connective by retrieving from the corpus pairs of consecutive sentences that do not contain a discourse connective linking them.

Initial data. *CETEMPúblico* (Rocha and Santos, 2000) is a corpus of Portuguese, built from excerpts of articles from *Público*, a Portuguese daily newspaper. It contains excerpts of articles belonging to different topics, including Politics, Sports, Opinion, Economy, etc. *CETEMPúblico* is split in 20 parts, divided in 2,571,735 paragraphs, containing 7,324,935 sentences with a total of 158,890,175 words.

Firstly, this corpus was annotated with LX-Suite (cf. *Annex B.1*). The corpus was then filtered and only the sentences that contain a verb were kept. The sentences without verbs can be titles or authors' names and, thus it would not be useful to have them in the discourse corpus. In this filtering process, 863,512 sentences, containing 698,408 words, were ignored. The table describing all the statistics for the initial corpus can be found in *Annex D.2.2*.

After being filtered, the corpus is prepared to be processed in order to retrieve the sentence pairs.

Discourse annotated corpus. Once the corpus has been defined and the discourse connectives have been identified (cf. Section 4.3.2.2), a set of pairs of sentences linked by a discourse connective was created.

Prasad et al. (2008) state that discourse connectives have typically two arguments: ARG$_1$ and ARG$_2$. Moreover, they concluded that the typical structure in which these three elements are combined is ARG$_1$ <CONNECTIVE> ARG$_2$.

Example 4.17 shows two sentences with this typical structure, where S_1 maps to ARG$_1$ and S_2 maps to ARG$_2$, with the connective *mas* ("but") being included in ARG$_2$.

Example 4.17

S_1	*Washington seguiu Saddam desde o início.* Washington followed Saddam from the beginning.
S_2	***Mas** a certa altura as comunicações com Clinton falharam.* **But** at some point communications with Clinton failed.

Therefore, our aim is to analyze the initial corpus to find each discourse connective and its arguments (ARG$_1$ and ARG$_2$), occurring in their typical structure ARG$_1$ <CONNECTIVE> ARG$_2$, where ARG$_1$ and ARG$_2$ can either be sentences or clauses, and <CONNECTIVE> is contained in ARG$_2$. Prasad et al. (2008) argue that this structure is the minimal amount of information needed to interpret a discourse relation.

Thus, considering this structure, a collection of triples instantiating both arguments and its relation was obtained. These triples are in the form: (*Sentence*$_1$, *Sentence*$_2$, *DiscourseRelation*), defining that *Sentence*$_1$ and *Sentence*$_2$ have *DiscourseRelation*.

In order to build the *PDTB*, Prasad et al. (2008) did this annotation by hand. In our case, the process of creating a Portuguese corpus of discourse annotated sentence pairs is done automatically.

Sentences from the filtered corpus (described above) that contain a connective from the list of connectives (cf. Section 4.3.2.2) are identified. At this point, these are only candidates to be part of the discourse corpus, since they have to meet two requirements to be added to the final dataset: (1) the cue phrase found must, in the context, be indeed a discourse connective, and (2) the sentence has to verify one of two conditions, based on the relative position of the connective present in it.

Concerning the first requirement, there are some cue phrases that are frequently ambiguous between a discourse connective meaning and several other meanings. As am-

biguity requires a specific approach, the handling of this problem is discussed below in Section 4.3.2.4.

Regarding the second requirement, related to the position of the connective, there are two ways to find such relations between two sentences: discourse connectives can either link two sentences together or two clauses. In the first case, the second sentence of the pair contains the discourse connective in its beginning. Otherwise, in the second case, a sentence contains a discourse connective in its middle, linking the two clauses.

Example 4.18 illustrates these two cases, that should be handled differently.

Example 4.18

S_1 *Washington seguiu Saddam desde o início.*
Washington followed Saddam from the beginning.
S_2 ***Mas** a certa altura as comunicações com Clinton falharam.*
But at some point communications with Clinton failed.

S_3 c_1=*A segurança do complexo chegou a ser feita por agentes da PSP,* c_2=***porém** hoje só existem os porteiros.*
The safety of the complex was initially performed by police officers, **however** today there are only the doorkeepers.

S_1 and S_2 are two sentences linked by the discourse connective *mas* ("but"), where S_1 maps to ARG_1 and S_2 maps to ARG_2 of the discourse relation. S_3, in turn, contains two clauses that are linked by the discourse connective *porém* ("however"). In this case, the first clause (c_1) maps to ARG_1, and the second clause (c_2) to ARG_2.

Handling sentences that start with a discourse connective. When the connective occurs in the beginning of ARG_2 of the discourse relation, and ARG_2 only contains one verb, both the sentence containing the connective (ARG_2) and the one that precedes it (ARG_1) are selected as a pair, as demonstrated in Example 4.18. So, that pair (S_1 =ARG_1, S_2 =ARG_2) is a valid sentence pair to be included in the final corpus.

If the connective occurs in the beginning of the sentence, and the sentence contains two main verbs, only this sentence should be considered to create the pair of sentences. Therefore, this sentence should contribute with the two arguments for the discourse relation, since each of its verbs define a clause, allowing the creation of two new well-formed sentences. Consider the sentence in Example 4.19.

Example 4.19

S_4	***Embora*** *não tenha ido às aulas, trabalhei bastante.*
	Although I have not attended the classes, I have worked very hard.

Since this sentence contains two main verbs *tenha ido* ("have not attended") and *trabalhei* ("have worked"), it has two clauses are the two arguments for the sake of the discourse relation. Therefore, the sentence is split in two sentences, considering the comma that divides both clauses, as described in Example 4.20.

Example 4.20

$S_{4_{c_1}}$	***Embora*** *não tenha ido às aulas.*
	Although I have not attended the classes.
$S_{4_{c_2}}$	*Trabalhei bastante.*
	I have worked very hard.

Given the canonical ARG_1 followed by ARG_2 adopted, the order of the sentences is inverted to obtain the sentence pair ($S_{4_{c_2}}$=ARG_1, $S_{4_{c_1}}$=ARG_2) to be entered into the discourse corpus as shown in Example 4.21.

Example 4.21

$S_{4_{c_2}}$	*Trabalhei bastante.*
	I have worked very hard.
$S_{4_{c_1}}$	***Embora*** *não tenha ido às aulas.*
	Although I have not attended the classes.

Handling a sentence that contains a discourse connective in its middle. Regarding the case where the discourse connective occurs in the middle of a sentence, as in Example 4.18 above, two types of sentences can be found: sentences with one verb or sentences with more than one verb.

If the sentence retrieved contains only one main verb, the connective is moved to the beginning of the sentence, forming the desired canonical structure. This sentence is thus the ARG_2 of the discourse relation, containing the connective in its beginning. Thus, this sentence will be considered as in the first step above. Hence, the sentence that precedes it (ARG_1) is selected and this pair is added to the list of sentence pairs.

Examples 4.22 , 4.23 and 4.24 illustrate this. Consider the initial sentence in Example 4.22.

Example 4.22

S_5	*O Governo Regional nega,* ***no entanto****, essa conclusão.* The Regional Government denies, **however**, this conclusion.

This sentence contains the discourse connective *no entanto* in the middle of the sentence and just one verb, *nega* ("denies"). The first step moves the discourse connective to the beginning of the sentence as shown in Example 4.23.

Example 4.23

$S5_{modified}$	***No entanto****, o Governo Regional nega essa conclusão.* **However**, the Regional Government denies this conclusion.

The next step aims to find the sentence that precedes this one in the corpus. This way ($S5_{previous}$=ARG$_1$, $S5_{modified}$=ARG$_2$) define a sentence pair, as illustrated in Example 4.24.

Example 4.24

$S5_{previous}$	*O custo de vida no Funchal é superior ao de Lisboa.* The cost of living in Funchal is higher than in Lisbon.
$S5_{modified}$	***No entanto****, o Governo Regional nega essa conclusão.* **However**, the Regional Government denies this conclusion.

When the sentence contains two main verbs, different arrangements should be made. Recall S_3 in Example 4.18. This sentence has two different clauses since it contains two main verbs: *chegou* ("was") and *existem* ("are"). In this case, the sentence is split in two where the discourse connective occurs and two sentences are obtained, defining a pair ($S3_1$=ARG$_1$, $S3_2$=ARG$_2$) to be included in the final corpus as shown in Example 4.25.

Example 4.25

$S3_1$	*A segurança do complexo chegou a ser feita por agentes da PSP.* The safety of the complex was initially performed by police officers.
$S3_2$	***Porém****, hoje só existem os porteiros.* **But** today there are only doorkeepers.

Classifying pairs of sentences. Considering each pair of sentences, they will now be classified with their corresponding discourse relation.

When studying the *PDTB* corpus looking for discourse relations, Pitler et al. (2008) showed that most of the discourse relations are explicitly marked by the use of a discourse connective. A discourse relation between two sentences is given by the class of the discourse connective that links them. The list of discourse connectives (cf. *Annex D.2.1*) defines for each connective a class, a type and a subtype. Consequently, the classification of the discourse connective (CLASS-TYPE-SUBTYPE) indicates the discourse relation between those two sentences. Thus, from a pair of sentences (*Sentence*$_x$, *Sentence*$_y$), a triple is obtained by using both sentences and their discourse relation, such as (*Sentence*$_x$, *Sentence*$_y$, CLASS-TYPE-SUBTYPE).

Consider the pair of sentences in Example 4.26.

Example 4.26

(*"O custo de vida no Funchal é superior ao de Lisboa."* =ARG$_1$,
"No entanto, o Governo Regional nega essa conclusão." =ARG$_2$)
("The cost of living in Funchal is higher than in Lisbon.",
"However, the Regional Government denies this conclusion.")

The first step of the classification process removes the discourse connective from the sentence defined as ARG$_2$. Then, the class of the discourse relation that links both sentences is retrieved. In this case, the class of *no entanto* ("however") is COMPARISON-CONTRAST-OPPOSITION, so the triple will include this class as the discourse relation between both sentences (Example 4.27).

Example 4.27

(*"O custo de vida no Funchal é superior ao de Lisboa."*,
"O Governo Regional nega essa conclusão.",
COMPARISON-CONTRAST-OPPOSITION)

This way a discourse annotated corpus has been built relating a pair of sentences and their respective discourse relation. The ambiguity issues concerning the classification of sentence pairs are addressed below in Section 4.3.2.4.

Unrelated sentences. When considering two adjacent sentences, we must also take into account that these sentences can share a discourse relation or not. So, for the sake of this task, during the creation of the corpus, pairs of adjacent sentences that are not linked by any of the connectives, have also been retrieved.

All these pairs were classified with the NULL class, stating that there is no explicitly conveyed relation between the two sentences.

Corpus distribution. Figure 4.3 displays the distribution of pairs stored for each class, (for full details about the distribution of the classes in the corpus see Section D.2.4 in *Annex D.2.2*).

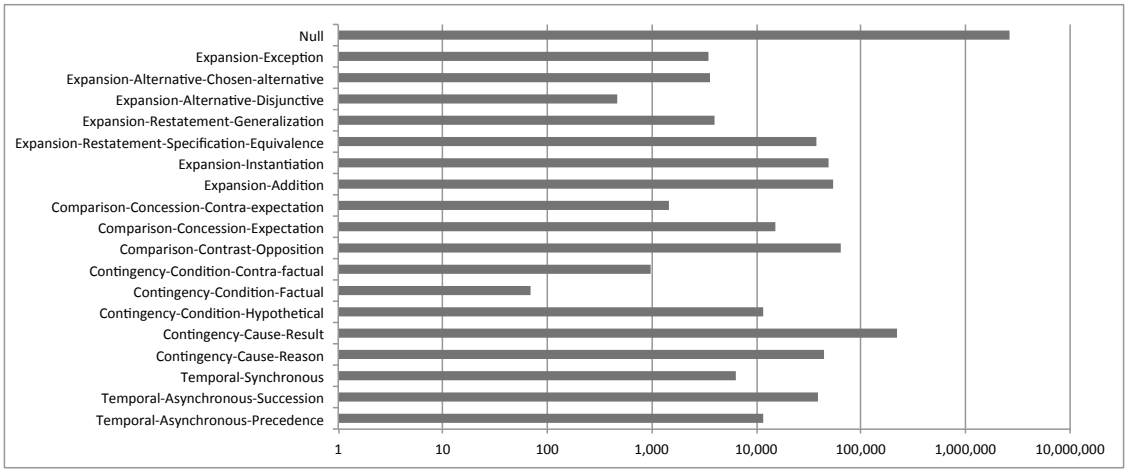


Figure 4.3: Distribution of the classes in the corpus.

A total of 3,237,608 pairs were extracted. As expected the NULL class is the largest portion of the corpus (82%), meaning that the vast majority of the sentences in the corpus do not contain a discourse connective along the heuristic rules defined above require.

When considering the corpus without the NULL class, there is a clearly dominant class – CONTINGENCY-CAUSE-RESULT – containing almost 40% of the pairs. At the same time some very small classes were found – e.g. CONTINGENCY-CONDITION-FACTUAL, EXPANSION-ALTERNATIVE-DIS-JUNCTION and CONTINGENCY-CONDITION-FACTUAL representing less than 1% of the classes in the corpus. This will drive the way the training of the classifier is performed.

Once the corpus containing the triples was obtained, it was ready to be used to train a classifier, that aims at recognizing discourse relations between two sentences.

4.3.2.4 Ambiguity

Ambiguity is a cross-cutting issue when dealing with this task. It is a twofold problem, as Pitler and Nenkova (2009) reported when studying the *PDTB* corpus. A connective can be ambiguous with respect to other semantic meanings, or it can be ambiguous across different discourse relations.

Pitler et al. (2008) argued that there are two productive ways to solve ambiguity: either by using syntactic rules or by selecting the most frequent occurrence of the connective, as it typically appears in its predominant sense. Miltsakaki et al. (2005) studied the construction of syntactic rules to distinguish different word meanings in *PDTB*, a solution that proved to be very effective.

In our work, both these solutions were used. Depending on the case, a connective was disambiguated either by selecting its most frequent class, or by building syntactic rules that seek to distinguish its meaning in context.

While building the **list of connectives** (cf. *Annex D.2.1*), both the above mentioned types of ambiguity were found.

After screening the extracted corpus to check if the pairs retrieved were correct with the support of an expert in Linguistics, we found out that there were many incorrect pairs, including cue phrases that have meanings other than being a discourse connective. The solution here was to remove these instances maintaining their more specific instances which are not ambiguous.

Consider for instance the adverb *depois* ("after"). This adverb can be used as a connective from the class TEMPORAL-ASYNCHRONOUS-SUCCESSION, but it is used more frequently as a temporal verb modifier. Example 4.28 shows examples for occurrences of *depois* and *depois de*.

Example 4.28

S_1 *Alguns nomes históricos do PS criticam as declarações de Guterres contra a liberalização,* ***depois de*** *terem votado a seu favor em 82.*
Some historical names from PS criticize the statements from Guterres against liberalization, **after** having voted for him in 82.

S_2 *Vês os desenhos animados* ***depois****.*
You can watch the cartoons **after**.

While in S_1, *depois de* is used as a connective linking both sentences in a TEMPORAL-ASYNCHRONOUS-SUCCESSION relation, in S_2 *depois* is used as a temporal modifier of the verb *vês* ("watch"). So, the instance *depois* was removed and its two more specific instances *depois que* and *depois de* were kept. The same occurred with the adverb *antes* ("before"), in class TEMPORAL-ASYNCHRONOUS-PRECEDENCE. It is used more frequently as a temporal modifier of a verb. So, *antes* was also removed and its more specific instances *antes de* and *antes que* were kept.

Also, there were some connectives that would belong to different classes or types or even subtypes. After a meticulous inspection of the corpus, these ambiguous connectives were placed in their predominant class.

Take the connective *ou* ("or"), for instance. The *PDTB* classification places this connective in two different subtypes of the class EXPANSION-ALTERNATIVE: CONJUNCTIVE and DISJUNCTIVE. Though the semantic nuance between these two subtypes is really hard to distinguish through automatic rules. While conjunctive means that two events can occur together, disjunctive means that they cannot occur together. Consider the sentences in Example 4.29.

Example 4.29

S_1 *Eu vou ligar ao João* ***ou*** *vou mesmo a casa dele.*
I will call John **or** I will indeed go to his place.
S_2 *O mercado imobiliário sobe* ***ou*** *desce.*
Either the real estate market rises **or** falls.

S_1 is an example of a CONJUNCTIVE *ou*. The two actions can be performed at the same level, so they both can be executed at the same time, though it is not possible to determine whether they are executed one after another or not. S_2 is an example of a DISJUNCTIVE *ou*. The two actions in this sentence cannot occur at the same time. As a semantic annotation was not available, we were not able to distinguish this two types of meanings for *ou*. So, in this case, we opted to merge these two subtypes in only one, named DISJUNCTIVE in further references.

Other cases, such as *se* ("if") that is ambiguous between all the subtypes of the class CONTINGENCY-CONDITION, were solved during classification (discussed below).

While **creating the corpus** by selecting the sentence pairs based on the list of connectives, ambiguity issues also arose. Here, a cue phrase has to be disambiguated in its

context, in order to verify if it is indeed being used as a discourse connective or not.

Heuristics embodied in pattern-matching rules were used to identify the occurrences in which these cue phrases are likely used as discourse connectives. The disambiguation rules for each connective are described in *Annex D.2.7*. In general, they define the structure the sentence must have before and after the connective appears.

Take the connective *por* ("by") as an example. This word can express more semantic values other than being a discourse connective: it can either introduce the "by"-agent in a passive or a modifier (preposition followed by a noun phrase). So, to make sure *por* is being used as a connective, it cannot be preceded by a verbal form (an inflected form, a past participle or a gerund) and must be followed by a verb in the infinitive. If a sentence containing *por* does not meet these two rules, it will not be used in the discourse corpus. After being applied, these rules determine if the sentence is to be included in the final corpus of discourse relations or not.

Consider, for instance, the sentence in Example 4.30.

Example 4.30

S_1 *O seu ensino foi introduzido em 1942, no Instituto Superior de Agronomia de Lisboa,* ***por Francisco Caldeira Cabral.***
His teaching was introduced in 1942, at the Institute of Agronomy in Lisbon, **by** Francisco Caldeira Cabral.

S_2 *E tantos foram os candidatos que o período destinado a testar a aplicação do RMG* ***acabaria por*** *ceder lugar a um processo efectivo de financiamento.*
And many were the candidates that the period for testing the application of RMG **would give way to** a process of effective funding.

S_3 *Muitas pessoas têm medo de aceitar novos desafios,* ***por*** *pensarem que não serão capazes de lidar com novas situações.*
Many people are afraid to accept new challenges, **because** they think they will not be able to deal with new situations.

In S_1, *por* is used as "by"-agent in a passive sentence, while in S_2 it is included in the construction of the verb *acabar* ("would give"), thus not complying with the first rule, that states that it cannot be immediately preceded by a verb inflected form. Yet, S_3 includes a valid occurrence of *por* as a connective. Note that in this case, *por* is followed by an infinitive form of a verb (*pensarem* – "think"), as the third rule states.

Ambiguity issues also arose when **classifying pairs of sentences**, as there are some connectives that remained ambiguous between different classes while building the list of connectives.

The connectives *desde que* ("as long as"), *se* ("if") and *caso* ("in case of") deserved special attention. The connective *desde que* can either be a CONTINGENCY-CONDITION-HYPOTHETICAL or a TEMPORAL-SYNCHRONOUS. Both *se* and *caso* were included in all the subtypes of CONTINGENCY-CONDITION: FACTUAL, HYPOTHETICAL and CONTRA-FACTUAL.

The solution here was to use heuristic rules in order to disambiguate these connectives (the same way as was done for *por*). Table D.32, Table D.33, and Table D.34 in *Annex D.2.8*, describe in detail the rules used to differentiate the subtype that must be assigned to each of these connectives, based on the verb expressed either in ARG_1 or in ARG_2 or in both.

Take *se*, for instance and its rules defined in Annex D.33. Examples 4.31 , 4.32 and 4.33 show pairs for each of its subtypes.

When considering the FACTUAL *se*, it must comply with two rules. Example 4.31 contains a pair that matches the first rule, as the verb in ARG_1 is in the indicative present (*é* – "is") and the verb in ARG_2 is in the indicative present (*tem* – "does").

Example 4.31: FACTUAL

ARG_1	*Não é nossa a culpa.*
	It is not our fault.
ARG_2	***Se** Casablanca não tem futuro.*
	If Casablanca does not have future.

The HYPOTHETICAL *se* must comply with a rule stating that the verb in ARG_1 is in the indicative form (*vão* – "will have") and the verb in ARG_2 must be in the subjunctive form (*vier* – "were to get").

Example 4.32: HYPOTHETICAL

ARG_1	*Os nossos adversários vão ter pesadelos.*
	Our opponents will have nightmares.
ARG_2	***Se** eu vier a conseguir juntá-los.*
	If I were to get them together.

Finally, the CONTRA-FACTUAL *se* must also cope with two rules. The first rule states

that the verb in ARG$_1$ should be in the indicative imperfect past or conditional (*ficaríamos* – "would be") and the verb in ARG$_2$ should be in the subjunctive imperfect past (*ultrapassássemos* – "did not overcome"), as shown in Example 4.33.

Example 4.33: CONTRA-FACTUAL

ARG$_1$	*Ficaríamos profundamente desiludidos.*
	We would be very disappointed.
ARG$_2$	***Se** não ultrapassássemos os 100 milhões de euros.*
	If we did not overcome over 100 million euros.

All the pairs in the previous examples were classified as CONTINGENCY-CONDITION. Afterwards, the disambiguation rules were applied in order to distinguish the subtype of the relation. Therefore, as demonstrated before, the pair of sentences was classified depending on the tense of the verb of the arguments. If the arguments do not verify these rules, the pair is ignored for the sake of the construction of the corpus of pairs. This way, it was possible to distinguish between all the subtypes considered.

4.3.2.5 Classifier training

The development of a classifier that decides which discourse relation holding between two sentences was the following step.

The distribution of discourse relations in the corpus (cf. Figure 4.3) indicates that it is highly uneven, containing some very big classes (viz. NULL) and at the same time some very small ones (e.g. CONTINGENCY-CONDITION-FACTUAL). Thus, the training procedure is accomplished in two phases. In the first training phase – *Nulls vs. Relations* –, the goal is to train a classifier that aims to identify whether the sentences entertain a discourse relation (different from NULL) or not. After uncovering that two sentences entertain indeed a discourse relation, the second training phase – *Relations vs. Relations* – seeks to find which is that discourse relation.

At this point, there are several decisions at stake: the features to be used, the training and testing datasets, the classification algorithm, etc. These decisions depend on each other, so that several experiments have been conducted using *Weka* workbench (Witten and Frank, 2005). In the next subsection we will cover the experimental settings on which the experiments performed were based, and which are described afterwards.

Experimental settings. In a classification procedure, there are decisions to make concerning the features, the datasets and the classification algorithms to be used.

Considering the task at hand, the **features** are expected to reflect properties that could be associated with the discourse relation between the two arguments of the relation.

In order to find the best configuration, several features were tested. Considering that the corpus is composed by pairs of sentences associated with a discourse relation, the most straightforward approach would be to used *both sentences* (ARG_1 and ARG_2) as features to train the classifier, even though the complete sentences are expected to have too much noise. Nevertheless, this was one of the options tested.

Previous works [(Marcu and Echihabi, 2002), (Lee et al., 2006), (Wellner et al., 2006), (Lin et al., 2009), (Louis et al., 2010)] experimented with different types of features to classify discourse relations, including contextual features, constituency parse features, dependency parse features, semantic features and lexical features.

Taking into account that our corpus is annotated with POS tags, we can neither use constituency or dependency parses nor semantic features. We decided thus to use lexical features. Therefore, our approach is inspired in the work by Wellner et al. (2006), that reported high accuracy when training classifiers with a combination of several lexical features.

In a sentence, the verb expresses the event so it can constitute a relevant information in helping to distinguish between different relations. Considering a specific relation, different pairs of sentences sharing that relation might have different verbs, although they could have the same discourse connective. This discourse connective typically requires the same range of verb inflections, not necessarily the same instance of the verb. Thus, instead of using the verb in each sentence as feature, we used the *verb inflection* feature of each argument.

Another feature resorted to is related to the context in which the discourse connective appears. Thus, a context window surrounding the occurrence of the discourse connective will be used. The corpus was built based on the assumption that the discourse connective in a discourse relation occurs in the beginning of ARG_2. Considering this, we used a three-word context window in the beginning of ARG_2 and a three-word context window in the end of ARG_1. This way we have a *six-word context window* surrounding the location where the discourse connective occurs (or could have occurred) in the discourse relation – recall from Section 4.3.2.3 that the discourse connective has been removed from ARG_2.

In addition, three more features were used to improve the identification of the small differences across discourse relations. These features include all the *adverbs*, *conjunctions* and *prepositions* found in each of the arguments of the discourse relation. Conjunctions link words, phrases, and clauses together. Thus, they can provide useful information about the way these are connected. Adverbs are modifiers of verbs, adjectives, other adverbs, phrases, or clauses. An adverb indicates manner, time, place, cause, or degree, so that it can unveil the grammatical relationships within the sentence or clause as a whole. A preposition may indicate the temporal, spatial or logical relationship of its object to the rest of the sentence. All these words can constitute clues to better identify the discourse relation between two unseen sentences, so they can help to enhance the accuracy of the classifier.

At this point, the corpus was split in two parts: a training part and a testing part. The **testing dataset** was created by selecting the fist 2500 pairs from the discourse corpus. Taking into account that the training procedure is carried out in two phases, two testing datasets were created, one for the first training phase *Null vs. Relations* (cf. Figure 4.4) and another for the second training phase *Relations vs. Relations* (cf. Figure 4.5). These datasets will remain unbalanced as to reflect the normal distribution of discourse relations in a corpus.

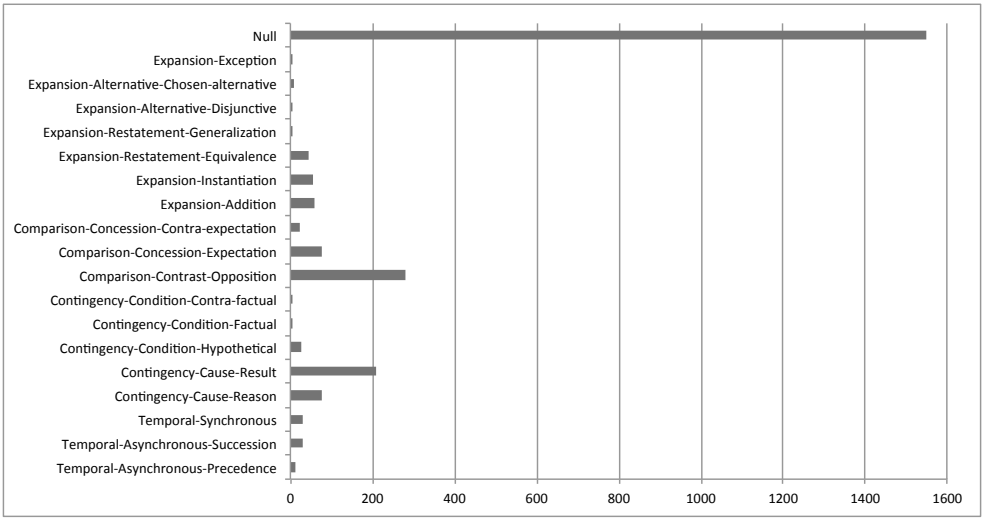


Figure 4.4: Distribution of the classes in the testing dataset for the first training phase – *Nulls vs. Relations.*

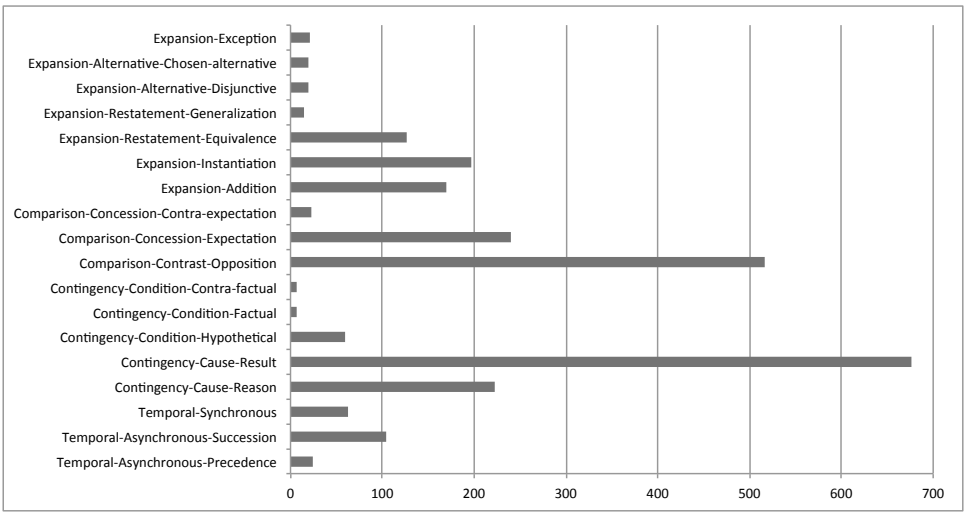


Figure 4.5: Distribution of the classes in the testing dataset for the second training phase – *Relations vs. Relations*.

In what concerns training datasets, their size and composition depend on the experiment being performed, so that the properties of both these datasets will be reported below when detailing the experiments. Note that the corpus is highly unbalanced, therefore all the experiments will rely on even training datasets. This means that the training datasets will always contain the same number of instances for each class, in order to improve the accuracy of the classifiers in finding the appropriate discourse relation.

There are several **classification algorithms** that have been more frequently used in Natural Language Processing tasks.

Naïve Bayes (John and Langley, 1995) is a simple probabilistic classifier. The algorithm assumes independence of features as suggested by Bayes' theorem. This means that a specific feature value can occur independently from any other feature value. Despite its simplicity, the results achieved in many application cases have been competitive with respect to the ones obtained with much more complex algorithms.

C4.5 (Quinlan, 1996) is a decision tree algorithm. It splits the data into smaller subsets using the information gain in order to choose the attribute for splitting the data. In short, decision trees hierarchically decompose the data, based on the presence or absence of the features in the search space.

Finally, *Support Vector Machines (SVM)* (Vapnik, 1995) is an algorithm that analyzes data and recognizes patterns. The basic idea is to represent the examples as points in

space, making sure that separate classes are clearly divided. More formally, the algorithm builds a hyperplane or a set of hyperplanes. The goal is then to find the hyperplane with the largest distance separating the vectors that represent the classes, that is different categories should be placed in different sides of the plane. *SVM* is a binary classifier, thus most suitable in classification problems involving two classes.

All these algorithms were used in the experiments reported.

Results. The classifier we are seeking to build aims to learn how to distinguish which is the discourse relation that two unseen sentences share if any.

The first assumption of such a task would be to take into account all classification classes at the same time. This means that, in this case, we must take into account 19 classes, including all the 18 possible discourse classes plus the NULL class. However, as the corpus, from which the datasets will be built, is highly imbalanced between all these 19 classes, the classifier training procedure, as stated above, was divided in two phases. In the first phase (*Null vs. Relations*), a classifier that aims to determine if two unseen sentences share a discourse relation or not will be trained. In the second phase (*Relations vs. Relations*), already knowing that both sentences share a discourse relation, another classifier will be trained to discover which relation is that.

All the results reported were obtained using *Weka*.

Null *vs.* Relations. The experiment procedure defines first the training and testing datasets used. The training dataset includes the same number of examples from all classes divided in a binary classification problem. So, the training dataset will be a balanced one, that is the number of instances of the NULL is identical to the number of instances of all the other classes together (named RELATION). The classifier will have then to decide if two sentences share a discourse relation ("yes") or if they do not ("no"). In the first experiment, it will contain 5,000 pairs evenly divided in both these two classes. Yet, the testing dataset contains 2,500 pairs that seek to reflect the normal distribution of the discourse relations in a corpus, so that it remains unbalanced, with a distribution as in shown in Figure 4.4.

Concerning the features, this first experiment aims to identify which ones support and improve the classifier accuracy.

Regarding the algorithms, the results were obtained using the most common classification algorithms used in Natural Language Processing tasks (described above).

The configuration for the first experiment is the following:

- Testing dataset – 2,500 pairs from Part#1;
- Training dataset – 5,000 pairs split in half for each class, NULL and RELATION.
- Features essayed:
  - Complete ARG_1 and ARG_2 sentences;
  - Verb inflections;
  - Verb inflections and 6-word context window;
  - Verb inflections, 6-word context window, adverbs, prepositions, and conjunctions (*inf-6cw-adv-prep-cj*).
- Algorithms:
  - Naïve Bayes;
  - C4.5 decision tree;
  - Support Vector Machines (SVM).

In order to understand the results obtained when varying the features and the algorithms, a baseline for this task must be considered. The baseline assigns the most frequent class to all the instances. By observing the classes distribution in the corpus (cf. Figures 4.3), we can see that the most frequent class is the NULL class. So, when assigning always the NULL class to the instances occurring in the test set, it is possible to achieve an accuracy of **61%**, being this value the lower bound to be overcome by a more sophisticated classifier.

The results for the first experiment are described in Table 4.1.

#	**Features**	***Naïve Bayes***	***C4.5***	***SVM***
1	*sentences*	59.00 %	57.84 %	63.68 %
2	*verb inflections* (*inf*)	57.56 %	58.84 %	60.64 %
3	+ *6-word context window* (*inf-6cw*)	59.12 %	60.16 %	61.32 %
4	+ *adverbs, prepositions and conjunctions* (*inf-6cw-adv-prep-cj*)	68.00 %	64.88 %	72.84 %

Table 4.1: Accuracy for each algorithm for the first experiment.

Taking into account the goal of this task – to find a discourse relation between two sentences –, the most straightforward approach, as was stated above, would be to use as features both sentences in ARG_1 and ARG_2 (feature#1). The first assumption when using this feature was that it might contain too much noise, as the whole sentences were being used. However, it might also contain some singularities that help it to achieve results close to the baseline. Feature#2 comprises all the verb inflections of the verbs in both sentences. By expressing an event, the verb can be a relevant source of information when regarding discourse relations and their specific inflections could help to identify the presence of a discourse relation or not. However, by itself this feature achieves results below the baseline when considering all the algorithms tested.

As suggested by Wellner et al. (2006), we essayed also to combine several features. When combining the verb inflections with the six-word context window – composed by three words in the end of ARG_1 and three words in the beginning of ARG_2 (feature#3) –, we were able to improve the accuracy with respect to all the three classifiers. With this configuration, it is even possible to overcome the baseline with *SVM*.

Finally, feature#4 includes the combination of the previous features with all the adverbs, prepositions and conjunctions found in both arguments. Using this combination, we were able to significantly improve the results of the three classifiers, with all scores overcoming the baseline.

When analyzing the behavior of each classifier, we can conclude that for all of them the combination of features keeps enhancing their accuracy. The best score obtained using *SVM* is the best one overall, being more than 10 percentage points above the baseline.

After finding a combination of features that overcomes the baseline with all the algorithms, a new experiment was performed. In this experiment, the size of the training dataset was the only variation.

The second experiment aims then to verify if extending the training dataset would improve the accuracy of the classifiers. Table 4.2 presents the results for the training dataset extensions.

The first line was obtained by training all the algorithms using a dataset containing 5,000 pairs. These are the final values reported in the previous experiment. We started then duplicating the training dataset until the learning curve has reached a point where no relevant improvements were obtained. The learning curve for each algorithm is displayed in Figure 4.6.

Number of pairs	***Naïve Bayes***	***C4.5***	***SVM***
5,000 pairs	68.00 %	64.88 %	72.84 %
10,000 pairs	67.80 %	67.88 %	75.20 %
20,000 pairs	67.68 %	69.76 %	76.72 %
40,000 pairs	67.08 %	70.56 %	76.80 %
80,000 pairs	67.52 %	72.96 %	78.68 %
160,000 pairs	66.96 %	70.20 %	78.92 %

Table 4.2: Accuracy when extending the training dataset for each algorithm.

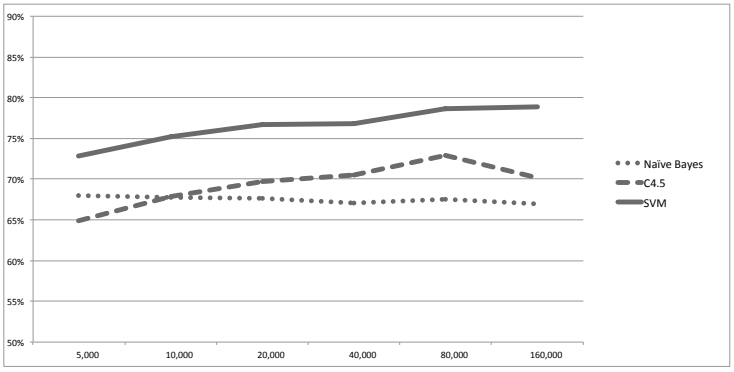


Figure 4.6: Learning curve when extending the datasets.

Note that two of the training algorithms keep performing better when doubling the training dataset until the 80,000 pairs mark. At this point, there are slight improvements (*SVM*) or even worse performances (*C4.5*). *Naïve Bayes* performs worst whenever the training corpus increases in size. As a result, the best performing algorithm – *SVM* – was used with a training dataset of 160,000 pairs to perform the first step of the connective insertion procedure: identify if two sentences enter a discourse relation or not.

Relations *vs.* Relations. Once the previous classifier has determined that the sentences entertain a discourse relation, it is necessary to identify which relation is that.

This is a multi-class classification problem, as we have 18 possible discourse relations to assign. Recall Figure 4.5 that shows the distribution of the classes in the testing dataset. Note that the most frequent discourse relation class is CONTINGENCY-CAUSE-RESULT. A baseline for this classification problem would assign the most frequent class to all the instances in the dataset, thus achieving an accuracy of **27%**, being this the lower bound to be overcome by a more sophisticated classifier.

As the combination of features used in the previous classifier (*Null vs. Relations*) proved to be effective, the same set of features was used in the current experiment (*inf-6cw-adv-prep-cj*). Also the algorithms tested were the same: *Naïve Bayes*, *C4.5* and *SVM*.

The first experiment takes together all the 18 classes in an *all-vs-all* approach. The training dataset contains 2,500 pairs split unevenly through all the classes. In the same way, the testing dataset contains 2,500 pairs aiming to reflect the distribution of the discourse relations in the corpus. Results from this experiment are reported in Table 4.3.

Classes	***Naïve Bayes***	***C4.5***	***SVM***
ALL CLASSES	22.64 %	23.36 %	29.6 %

Table 4.3: Accuracy for all the classes using a *all-vs-all* approach.

As these results point out, deciding between 18 different classes in a single step is a very hard task. Even though the best result (*SVM*) is slightly above the baseline, this is a disappointing outcome.

After looking at these results and taking into account that the best performing algorithm – *SVM* – is specially suitable for binary classification, this problem has been split in several problems, assuming a *one-vs-all* approach. Hence, for each class, we trained a classifier that aims to determine if a given pair of arguments entertains that relation or any of the other relations. This way, we turned a multi-class problem into multiple problems of binary classification.

The next experiment was based on training datasets containing 2,500 pairs divided in two: 1,250 from the specific relation at stake and 1,250 from all the other relations. The goal was to build training dataset in each sub-experiment with identical number of instances for the class at stake. However, it was not possible to obtain in the corpus 1,250 training instances for three classes (CONTINGENCY-CONDITION-FACTUAL, EXPANSION-ALTERNATIVE-DISJUNCTIVE and CONTINGENCY-CONDITION-CONTRA-FACTUAL). For each of these classes, we built the training dataset by including the maximum number of instances for it (66, 458 and 927, respectively) and that same number of instances for the other class altogether.

Thus, the training datasets for these three classes contained a total of 132, 916 and 1854, respectively.

Table 4.4 details the accuracy values obtained when training a single classifier for each class, using training datasets containing 2,500 pairs.

Classes	***Naïve Bayes***	***C4.5***	***SVM***
CONTINGENCY-CONDITION-FACTUAL	47.31 %	60.96 %	61.43 %
EXPANSION-ALTERNATIVE-DISJUNCTIVE	59.40 %	56.45 %	65.38 %
CONTINGENCY-CONDITION-CONTRA-FACTUAL	79.04 %	76.97 %	80.83 %
COMPARISON-CONCESSION-CONTRA-EXPECTATION	61.36 %	68.66 %	89.26 %
EXPANSION-EXCEPTION	56.77 %	61.36 %	74.13 %
EXPANSION-ALTERNATIVE-CHOSEN-ALTERNATIVE	62.79 %	61.88 %	64.87 %
EXPANSION-RESTATEMENT-GENERALIZATION	55.25 %	59.19 %	70.53 %
TEMPORAL-SYNCHRONOUS	62.87 %	57.29 %	62.23 %
TEMPORAL-ASYNCHRONOUS-PRECEDENCE	68.66 %	62.46 %	89.74 %
CONTINGENCY-CONDITION-HYPOTHETICAL	73.61 %	74.29 %	77.40 %
COMPARISON-CONCESSION-EXPECTATION	66.03 %	62.04 %	75.48 %
EXPANSION-RESTATEMENT-SPECIFICATION-EQUIVALENCE	87.51 %	94.97 %	71.89 %
TEMPORAL-ASYNCHRONOUS-SUCCESSION	83.43 %	70.54 %	76.60 %
CONTINGENCY-CAUSE-REASON	62.44 %	59.56 %	60.51 %
EXPANSION-INSTANTIATION	62.63 %	60.88 %	75.80 %
EXPANSION-ADDITION	59.72 %	59.76 %	89.34 %
COMPARISON-CONTRAST-OPPOSITION	57.29 %	57.60 %	69.18 %
CONTINGENCY-CAUSE-RESULT	62.44 %	57.00 %	62.83 %

Table 4.4: Accuracy for each class using a *one-vs-all* approach.

The results shown in the Table point out that all the classifiers performed significantly better when compared to the *all-vs-all* experiment. Moreover, by using a *one-vs-all* approach, we were able to create classifiers for each class. These are not only highly above the baseline but are also able to distinguish a specific class from all other classes.

As expected from previous results, *SVM* was the best performing algorithm, as it is specially suitable for binary classification.

Despite having small datasets for CONTINGENCY-CONDITION-FACTUAL, EXPANSION-ALTERNATIVE-DISJUNCTIVE and CONTINGENCY-CONDITION-CONTRA-FACTUAL, the classifiers achieved good results in distinguishing between all the other classes and these very small ones, attaining interesting results in these very specific cases.

On the other end, there are some very big classes (e.g. CONTINGENCY-CAUSE-REASON and CONTINGENCY-CAUSE-RESULT) that despite there being many instances for training it

is still hard to distinguish between these classes and all others. This also suggests that having more examples to train might not add much value if more helpful features are not used. Still, in this experiment, the same set of features was used in order to validate the comparison between the accuracy of all classifiers.

It is also relevant to note that, considering *SVM*, almost half of the classifiers perform above 75%, indicating that there are very accurate decisions being taken. This means that the singularities of these relations are well expressed by the set of features that was chosen. Moreover, it enhances the possibility of deciding correctly when inserting a connective between two raw sentences.

Taking into account that *SVM* had the best performance in this experiment, the classifiers used in the connective insertion procedure were trained using *SVM*.

4.3.2.6 Connective insertion

The classifiers described above were used to support a tool that inserts discourse connectives. The connectives were inserted only between the sentences in the paragraphs previously defined (cf. Section 4.3.1). Figure 4.7 outlines this procedure.

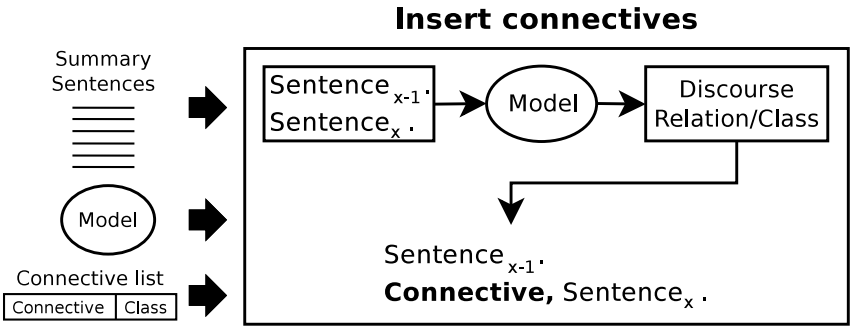


Figure 4.7: Connective insertion overview.

Recall that a paragraph is composed by a set of sentences. This procedure aims to insert a discourse connective between the sentences composing each paragraph, wherever a discourse relation is found. So, for adjacent sentences in every pair, firstly, we need to find out if they entertain a discourse relation or not. In order to do this, the *Null vs. Relations* classifier is applied.

At this point, the order between the sentences in the paragraph is not definitive. So, this classifier is applied both in the original order of the sentences (ARG_1 followed by ARG_2)

and in their inverted order (ARG$_2$ followed by ARG$_1$). If a discourse relation is not found in neither of these orders, no discourse connective will be inserted between these sentences, and they will be included in the paragraph in their original order. Otherwise, if a relation is found in any of these orders, the classifier *Relations vs. Relations* is then applied to uncover the discourse relation between those sentences. This classifier returns the CLASS-TYPE-SUBTYPE triple with the highest confidence score, thus setting the discourse relation between these sentences.

Afterwards, the list of connectives (detailed in Section 4.3.2.2) is searched for all the connectives that pertain to that specific class. One of these connectives is then retrieved from this list randomly, if it has not been already assigned to any other pair of sentences. It is inserted in the beginning of the second sentence, complying with the typical structure ARG$_1$ <CONNECTIVE> ARG$_2$ – if the order of the sentences has been changed during the first stage of the process, this structure is thus inverted.

Example 4.34 illustrates this process.

Example 4.34

1. Two adjacent sentences.
"O custo de vida no Funchal é superior ao de Lisboa."
"The cost of living in Funchal is higher than in Lisbon."
"O Governo Regional nega essa conclusão."
"The Regional Government denies this conclusion."

2. Finding the discourse relation.
After applying model *Null vs. Relations* → "Yes" = both sentences entertain a non NULL discourse relation.
After applying model *Relations vs. Relations* → relation class = COMPARISON-CONTRAST-OPPOSITION

3. Looking for the connective to insert.
A random connective is obtained in the list for the class COMPARISON-CONTRAST-OPPOSITION → retrieved: *"no entanto"* (however).

4. Inserting the discourse connective between the two sentences.
"O custo de vida no Funchal é superior ao de Lisboa."
"The cost of living in Funchal is higher than in Lisbon."
"No entanto, o Governo Regional nega essa conclusão."
"However, the Regional Government denies this conclusion."

After the connective insertion process has been applied to every paragraph, the final summary is ready to be delivered.

Recalling our example, Table C.13 shows the sentences after the paragraphs have been created. Table C.14 (in *Annex C*) shows the final summary that is delivered to the end user after the connective insertion procedure has been applied to the sentences within each paragraph. In the first step, the classifier *Null vs. Relations* determined that sentence#3 and sentence#4 shared a discourse relation. Then, in the second step of the procedure, the classifier *Relations vs. Relations* assigned those sentences a discourse relation of CONTINGENCY-CAUSE-RESULT. By randomly selecting between all the possible connectives in this class, the connective inserted was *assim* (thus).

Inserting connectives between related sentences is a task that can help to improve the textual quality of a summary, whose content was retrieved from different texts, thus without any known relation.

4.4 Discussion

The post-processing phase includes several steps. All decisions concerning the three stages in the post-processing module were carried out considering the process as a whole. The order of application of the procedures has been defined based not only on the propose of each step, but also on the outcome produced.

Thus, compression tasks need to be executed first, as they define the sentences in the summary. Condensing sentences – sentence reduction – makes room for further sentences to be included in the summary. The structures targeted to be removed are expected to contain elaborative information. By removing these structures, this information is thus replaced by other information through the inclusion of new (reduced) sentences in the text – compression review –, taking always into account the compression rate to be met. Prior to the sentence reduction process, sentences were ordered by relevance order. This is the reason why compression tasks must be the first step of the post-processing, as the other steps require that the final set of sentences to be included in the summary has already been defined.

Then, fluency tasks are applied in sequence. Paragraph creation addresses the ordering problem of a multi-document summary. The main goal for this task is to help improve the readability of the summary. By using the information obtained through the clustering

by keywords procedure (cf. Section 3.3.2), sentences that share the same keywords are grouped together in the same paragraph. Also, their order within the paragraphs must be considered. In this case, a relevance score was used to determine the order of the sentences in the paragraphs. A final decision had to be made concerning the order of the paragraphs in the summary. The approach undertaken defined that the highest scored paragraphs would appear first, as the highest scored sentences would contain the most relevant information.

After paragraph creation comes the connective insertion procedure. Discourse connectives are inserted only between the sentences composing a paragraph, and only if a discourse relation was found. Despite this step includes new words in the summary, this should not be considered a problem, as discourse connectives rarely contain more than two words. In fact, enhancing the textual quality of the summary was considered more important than not meeting the exact compression rate.

Considering the post-processing process as a whole, the assumption was that the key content of the summary would have to be defined first – compression tasks. Afterwards, taking into account this content, the structure of the text – paragraph creation – would have to be defined, in order to finally improve its cohesion – connective insertion – and thus its overall textual quality.

Following the example used in this dissertation, Example 4.35 shows the final summary retrieved, built using SIMBA, along with the post-processing module.

Example 4.35

Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.

O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais. Assim, todos morreram quando o avião não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.

This summary contains not only sentences that were simplified, but also it is split in paragraphs which contain discourse connectives that were inserted automatically (e.g. *assim* in the third sentence). This is the type of summary delivered to the end user after running SIMBA and applying the post-processing module.

4.5 Summary

In this Chapter, the post-processing module was described, by detailing all the steps that define both compression and fluency enhancement tasks. At the point where the post-processing step is applied, the sentences for the summary have been selected, ordered and extracted complying with the compression rate defined by the user.

The first task in the post-processing, compression, takes these sentences and reduces them to their key content – sentence reduction. Afterwards, as the desired compression has been compromised, the compression review step adds new sentences to the summary in order to make sure the compression rate is met again, taking these sentences from the ordered set of sentences retrieved from the source texts. These two steps are applied iteratively until there are no more sentences to be reduced, so that no more sentences can be added to the summary. By iterating between these two steps, it is possible not only to make sure that the desired size of the summary has been achieved, but also that new information can be added to the summary, while superfluous one is removed from the extracted sentences.

The result of the compression tasks is the input to the fluency tasks. Firstly, the new set of reduced sentences is arranged into paragraphs in order to group them in topics. By making use of the result of the clustering by keywords procedure, each cluster defines a paragraph that will contain the sentences selected to figure out in the summary. Then, between the sentences composing each paragraph, discourse connectives have been inserted in order to create discourse connections between the sentences in the paragraphs. This is therefore the summary delivered to the user.

CHAPTER 5

EVALUATION

Evaluation of an automatic summarization system is a challenging task. It may require manually build summaries to be compared to the summaries produced automatically, in order to obtain performance scores. Or, human users can be asked to express their judgment about the automatic summaries. Both these approaches depend ultimately and heavily on humans to be accomplished.

In the approach to automatic summarization developed in our work, this is even more acute, as the post-processing process adds sophistication to the textual quality of the output summary and thus further challenges to the evaluation of the final summaries. By modifying the text to enhance textual quality in a way that current automatic metrics may not detect, our approach to automatic summarization calls for a process of human evaluation to be fairly assessed.

Therefore, two types of evaluations were performed in order to judge the quality of SIMBA. On one hand, an automatic evaluation – detailed in Section 5.2 – was accomplished by using state-of-the-art automatic measures. On the other hand, a human evaluation procedure – discussed in Section 5.3 – was carried out by asking human users to rate the summaries created automatically. The results obtained in both these procedures are discussed in Section 5.4. This Chapter starts though by an overview of the corpus used in the evaluation procedures (Section 5.1).

5.1 Corpus

This Section details the corpus used in both the automatic and human evaluation procedures and in the statistics around the two main tasks executed by SIMBA: the summarization strategy and the post-processing module.

The evaluation process uses the only corpus available for the evaluation of multi-document summarization in the Portuguese language, *CSTNews* (Aleixo and Pardo, 2008). *CSTNews* is an annotated corpus composed by texts in Portuguese built to aid the development of a multi-document parser. Each set contains in average three documents which address the same subject. These texts were collected between August and September 2007 from five Brazilian on-line newspapers: *Folha de São Paulo*, *Estadão*, *O Globo*, *Jornal do Brasil* and *Gazeta do Povo*. In addition, for each set of texts, the corpus contains an ideal summary written by hand by a human abstractor.

This corpus contains 50 sets of news texts from several domains, making a total of 140 documents with 2,247 sentences and 47,434 words. Each set contains, on average, 3 documents addressing the same subject, where each contains on average 16 sentences and 339 words. Concerning the ideal summaries, these contain a total of 6,859 words making an average of 137.18 words per summary. The average compression rate of the ideal summaries in this corpus is 85%.

CSTNews will be used in both evaluation procedures. In the automatic evaluation, the whole corpus is used. In the human evaluation, a smaller subset was selected.

Statistics

Prior to the evaluation results, we present some statistics concerning the two main differentiating factors of this approach, the double-clustering phase and the post-processing procedure. The aim here is to fully understand the main differences between the summaries that are being compared during the evaluation procedures.

Content selection. In order to understand the impact of the double clustering approach, Table 5.1 details the number of sentences considered before and after the clustering phases have been executed, by taking into account their number of sentences and their number of words.

Clustering by similarity:			**Clustering by keywords:**		
Before	After	Difference	Before	After	Difference
2,247	2,115	132	2,115	1,599	516

Table 5.1: Sentences involved in the clustering phases.

The similarity clustering is the first step of the summarization process, so it takes as input all sentences in the corpus (2,247). After this step, 132 sentences have been ignored. These sentences correspond to 5% of the sentences in the corpus and are not considered in the following steps of the summarization process.

After the set of sentences has been filtered by means of the similarity clustering, the remainder of the sentences (2,115) are clustered by keywords.

There were 516 sentences that were added to the no-keyword cluster, meaning that they do not convey the most relevant information. This corresponds to 24% of the sentences in the corpus. Therefore, only 76% of the sentences taken into account in the keywords clustering phase were considered relevant to be retained.

This way, after executing the double clustering procedure, a total of 648 sentences, that is 29% of the corpus, were discarded. Note that when there is a compression rate that requires more sentences to be added to the summary, the sentences in this list of discarded sentences are used to complete the summary.

The goal of this procedure is twofold. On the one hand, it seeks to avoid that the summaries contain repeated information. On the other hand, it seeks to circumscribe information that may be considered relevant given the topics of the input set of texts.

Post-processing. The post-processing module performs two possible types of modifications in the text. While sentence reduction removes words from the summary, both compression review and connective insertion add words to it.

This study was performed over the summaries that were used in the evaluation procedure. This means that all the values were obtained for candidate summaries with a compression rate of 85%, as this is the compression of the ideal summaries in the *CST-News* corpus (cf. Section 5.1). Thus, the number of words being considered composes only 15% of the initial document sets.

Considering the post-processing procedure as a whole, Table 5.2 shows the number of sentences and words taken into account before and after this stage has been applied.

Before post-processing:

	Sentences	**Words**
Total	248	7,748
Average per summary	4.96	154.96

After post-processing:

	Sentences	**Words**
Total	285	7,907
Average per summary	5.7	158.14

Table 5.2: Sentences involved before and after post-processing is applied.

Before being post-processed, the candidate summaries have a total of 248 sentences, containing 7,748 words, representing 11% of the sentences in the corpus. After post-processing has been performed, the final summaries have a total of 285 sentences, with 7,907 words, representing 13% of the sentences in the corpus. The candidate summaries had an average of 5 sentences, while the final summaries will have 6 sentences on average. This means that the post-processing procedure enabled the insertion of one sentence on average in each summary. Recall that the compression review and the connective insertion procedures add words to the summary. Consequently, the number of sentences that can be included in the summary after the post-processing has been applied takes also into account the number of words that are inserted by both these procedures.

Table 5.3 reports on the specific number of words involved in each post-processing step (for further details see Section E.2 in *Annex E*).

	Sentence reduction	**Compression review**	**Connective insertion**
Total	-205	299	65
Average per summary	-4.1	5.98	1.13

Table 5.3: Words added or removed in the post-processing tasks.

A total of 205 words (9% of the corpus) have been removed by reducing sentences to their main content, through the sentence reduction process. In the opposite direction, 364 words (16% of the corpus) were inserted in the texts when executing both compression review and connective insertion.

Summing up, sentence reduction makes room for more 205 words of new information. Even though, connective insertion introduces a total of 65 functional, content-empty words, there is still room for 140 content words. When comparing this value to

the average length of the ideal summaries (137.18 words), it is possible to see that there is plenty information that can be included in the summary. Note that this study only considers summaries composed by 15% of the sentences in the corpus. In fact, when constructing larger summaries, the impact of these procedures can be even higher.

Finally, Table 5.4 details the number of connectives inserted by class.

	Total instances	**Total words**
COMPARISON-CONCESSION-CONTRA-EXPECTATION	2	1
COMPARISON-CONTRAST-OPPOSITION	5	8
CONTINGENCY-CAUSE-REASON	2	3
CONTINGENCY-CAUSE-RESULT	3	8
CONTINGENCY-CONDITION-HYPOTHETICAL	1	2
EXPANSION-ALTERNATIVE-DISJUNCTIVE	3	3
EXPANSION-EXCEPTION	4	4
EXPANSION-INSTANTIATION	3	5
EXPANSION-RESTATEMENT-GENERALIZATION	1	3
EXPANSION-RESTATEMENT-SPECIFICATION-EQUIVALENCE	3	6
TEMPORAL-ASYNCHRONOUS-PRECEDENCE	2	2
TEMPORAL-ASYNCHRONOUS-SUCCESSION	9	15
TEMPORAL-SYNCHRONOUS	2	4
Total	40	64

Table 5.4: Classes of the connectives that have been inserted in the summaries.

Recall that when the discourse dataset was created, the distribution of the classes was not even (cf. Figure 4.3). By being the most frequent class, CONTINGENCY-CAUSE-RESULT was expected to be the most assigned class. However, this is not the case. In fact, there is a very broad set of classes that have been inserted. In 18 possible classes, 13 different types of connectives were insert, possibly suggesting that the extraction of sentences permits to select a diversified set of pairs and discourse relations and the classifiers are able to detect the differences between all the classes.

After the statistics of the impact of the post-processing procedure have been analyzed, results of evaluation were obtained for the summaries generated by SIMBA. The following section reports on these results.

5.2 Automatic evaluation

Automatic summary evaluation is typically based on metrics concerning the overlapping of strings in the reference and the under evaluation summaries. These metrics seek to evaluate if the information that should be in the summary is indeed present, by comparing automatic summaries with ideal summaries made by human users. This means that such metrics evaluate an automatic summary based on what the human summarizers found more relevant to be in the ideal summary.

The most common metric used is ROUGE. Despite automatic evaluation is not expected to be fair when comparing the summaries produced by SIMBA with other summaries, acquiring these evaluation metrics still enables the comparison between the approach proposed in this work and other approaches. That is possible given the current state of the art in terms of evaluation of automatic summarization.

Hence, this Section reports on the automatic evaluation of SIMBA. The performance of several versions of SIMBA – obtained by adding or removing some of its modules – were compared against baselines specially created for this evaluation. Also, a comparison with the only summarization system available for the Portuguese language, GISTSUMM (described in Section 2.1.7), was carried out.

The remainder of this Section is organized as follows: Section 5.2.1 overviews the automatic evaluation procedure, the metrics used and the compression rate used in the experiments. Afterwards, Sections 5.2.2 , 5.2.3 , 5.2.4 and 5.2.5 report on the evaluation of SIMBA. Finally, Section 5.2.6 discusses the results of the automatic evaluation process.

5.2.1 Procedure

The automatic evaluation process described in this Section aims to evaluate all the modules of SIMBA in separate procedures. In general, this process is based on the comparison of each module of SIMBA with a baseline system, which can be a naïve approach to the task. This way, two types of automatically generated summaries are being compared: a so-called "baseline summary" and a summary produced by the system or module being evaluated – the "module summary".

The following subsections detail the evaluation for each module. Firstly, the baselines are described. Then, the results, obtained using ROUGE, that compare both the "baseline summary" and the "module summary" with the ideal summaries are discussed.

Metric. The summaries produced were automatically evaluated against the ideal summaries with ROUGE (cf. Section 2.1.5) by computing precision, recall and f-measure metrics. These metrics are intended to indicate the proximity between the automatic summary and its corresponding ideal summary. It is important to note that the closer to 1 the values are, the better the summary is. In fact, a more fine-tuned metric of ROUGE was used, *ROUGE-L* (longest common subsequence). The post-processing procedure introduces gaps in the extracted sentences. This metric does not require consecutive matches but in-sequence matches, which reflect the word order at sentence level. Thus, this is a fairer metric considering the type of arrangements that are made in the text. The final score obtained with ROUGE is an average of individual scores – precision, recall and f-measure – of the summaries generated by the summarizer.

Compression rate. The summaries were built over the texts included in the multi-document corpora *CSTNews*. As this corpus contains 50 sets of documents, there are 50 summaries produced automatically by each system. In all the experiments, a compression rate of 85% was used, meaning that the summary is 15% of the size of the set of input texts. This was the value chosen for the compression rate because the ideal summaries contained in the corpus have an average compression rate of 85%. Using the same compression rate of the ideal summaries allows a more accurate comparison not only between both automatic summaries, but mainly between the ideal summaries and the automatically generated summaries.

5.2.2 Sentence reduction

In order to perform the automatic evaluation of the sentence reduction process, for each set of documents in the corpus, two types of summaries were created. Firstly, a summary built by the sentence reduction baseline (described below), and, afterwards, a summary built by running SIMBA's summarization process, including only the sentence reduction step of the post-processing procedure.

Baseline. A baseline for sentence reduction can be the removal from each sentence of all its targeted structures. This algorithm, BLIND REMOVAL, is described in Section 4.2.1.2

and accomplishes this. It removes from a sentence all its parenthetical phrases, appositions, sentence-level prepositional phrases, and appositive relative clauses.

Hence, a summary-to-be is built by the main summarization procedure (cf. Chapter 3), that is the sentences are selected and the compression rate is applied. The sentence reduction then takes the collection of sentences in the summary-to-be, identifies its removable passages, and removes them all, retrieving a reduced summary. No further post-processing procedures have been applied in this case.

Results. Table 5.5 reports on the *ROUGE-L* values obtained for the baseline and SIMBA only with sentence reduction (REDUCED).

	BASELINE	**REDUCED**
Precision	0.4852	0.4479
Recall	0.4455	0.5057
F-measure	0.4588	0.4702

Table 5.5: Sentence reduction automatic evaluation.

The main difference between these summaries lies in the reduction algorithm, as the summarization process is the same. When observing the scores in Table 5.5, we can conclude that reduced summaries have a better overall performance when compared to the summaries produced by the baseline. This is a direct consequence of applying the reduction algorithm detailed in Section 4.2.1. These results point out that having a criterion that guides the reduction is relevant. In fact, the baseline system has removed relevant information from the sentences which justifies its worse scoring. The algorithm used in the REDUCED version of SIMBA, in turn, attempts to make sure that no key information is removed. As the two systems have the same underlying extraction procedure, the reduction process constitutes their differentiating factor.

5.2.3 Paragraph creation

In order to perform the automatic evaluation of the paragraph creation process, for each set of documents in the corpus, two types of automatic summaries were created. Firstly, a summary built by the paragraph creation baseline (described below), and afterwards, a summary built by running SIMBA's summarization process, including only the paragraph creation step as a post-processing procedure.

Baseline. The paragraph creation baseline takes as input a set of sentences. A random number x (where $2 \geq x \leq 6$) is obtained defining how many sentences will be included in each paragraph. Then, x sentences are grouped in each paragraph, in their original order, until all the sentences have been considered.

Results. Table 5.6 reports on the *ROUGE-L* scores obtained for the baseline and SIMBA only with the paragraph creation module (WITH PARAGRAPHS).

	BASELINE	WITH PARAGRAPHS
Precision	0.4483	0.4460
Recall	0.4947	0.5010
F-measure	0.4648	0.4669

Table 5.6: Paragraph creation automatic evaluation.

The main difference between the summaries being evaluated is the way they are organized. Both summaries contain the same sentences though ordered in different ways. For the baseline, the sentences are assigned to each paragraph based on its size. For the SIMBA's version of the system, the sentences intend to be organized in a way that the sentences that compose each paragraph are related among themselves by being related to a shared topic. By containing the same words, the scores for these systems are expected to be very similar. When observing Table 5.6 this expectation can be confirmed.

Nevertheless, in a very interesting way, *ROUGE-L*, a more granular metric, is able to detect some differences between the summaries, as, in some cases, the order of the words is different. Scores show that the summaries created by SIMBA's paragraph creation module are closer, by a very small difference, to the ideal summaries than the ones of the baseline, when considering both f-measure and recall. This means that the summaries from SIMBA's paragraph creation module contain a higher proportion of words in the same order as the ones in the ideal summaries.

5.2.4 Connective insertion

In order to perform the automatic evaluation of the connective insertion process, for each set of documents in the corpus, two types of summaries were created. Firstly, a summary built by the connective insertion baseline (described below), and afterwards, a summary

built by running SIMBA's summarization process, including both the paragraph creation and the connective insertion steps in the post-processing procedure.

Baseline. The connective insertion baseline takes as input a set of paragraphs. It inserts connectives in the same way as in the post-processing step, that is between sentences, from the second sentence until the last sentence, in each paragraph. However, the selection of the connective here is performed randomly. This means that taking into account all the connectives available (from all classes), a single connective is randomly selected and inserted in the beginning of each sentence in the paragraph.

Results. Table 5.7 reports on the *ROUGE-L* values obtained for the baseline and SIMBA with connective insertion module (CONNECTIVES).

	BASELINE	CONNECTIVES
Precision	0.4412	0.4438
Recall	0.4984	0.5013
F-measure	0.4626	0.4657

Table 5.7: Connective insertion automatic evaluation.

The results reported in Table 5.7 suggest two sets of very similar summaries. Although, and interestingly equivalent to paragraph creation, the summaries created using SIMBA's algorithm achieve slightly better results, specially when considering f-measure – the harmonic mean of precision and recall. This means that, in an overall assessment, SIMBA's summaries are closer to the ideal summaries than the baseline summaries, as of correctness and relevance of the words they contain. As *ROUGE-L* is a more granular metric these small differences can be detected.

5.2.5 Summarization

Several scores were obtained for the overall summarization procedure. The first phase compares the summaries created by SIMBA with the ones created by random baseline. The second phase compares SIMBA's summaries with the ones produced by GISTSUMM. In the third phase, the post-processing procedure of SIMBA is under scrutiny, so both summaries with and without post-processing are put together for comparison.

Baseline evaluation. The baseline created for this purpose takes all the sentences in the input collection of texts and randomly selects sentences up to making the number of words defined by the compression rate previously stated.

Table 5.8 shows the *ROUGE-L* results for SIMBA and this BASELINE.

	BASELINE	**SIMBA**
Precision	0.3971	0.4463
Recall	0.4271	0.5064
F-measure	0.4068	0.4698

Table 5.8: Baseline evaluation.

As shown in the Table above, SIMBA outranks the baseline system. These results suggest that randomly selecting the sentences to be included in the summary is not the best approach to summarization. By taking a more sophisticated approach we were able to improve the scores of a random system by six percentage points.

GISTSUMM evaluation. At this point, the summaries produced by SIMBA are being compared with the ones produced by GISTSUMM. Table 5.9 reports on the *ROUGE-L* values obtained for both these systems.

	GISTSUMM	**SIMBA**
Precision	0.4339	0.4463
Recall	0.3823	0.5064
F-measure	0.4017	0.4698

Table 5.9: SIMBA vs GISTSUMM evaluation.

As shown in Table 5.9, SIMBA outranks GISTSUMM. The difference between GISTSUMM and SIMBA f-measure values is of almost seven percentage points, which is a considerable difference. Both SIMBA's precision and recall scores also overcome the GISTSUMM scores. The precision value attained by SIMBA is very interesting. It suggests that the information contained in SIMBA's summaries is more accurate. The summaries include the key information conveyed by the ideal summaries, preserving, this way, the gist of the input collection of texts. Moreover, SIMBA's summaries have an higher recall value than the one by GISTSUMM, meaning that SIMBA summaries cover more topics mentioned in the input texts.

Post-processing evaluation. Table 5.10 reports on the *ROUGE-L* scores obtained for SIMBA with and without post-processing.

	WITHOUT POST-PROCESSING	WITH POST-PROCESSING
Precision	0.4460	0.4463
Recall	0.5010	0.5064
F-measure	0.4669	0.4698

Table 5.10: Post-processing evaluation.

In what concerns the comparison between summaries produced by SIMBA either WITH or WITHOUT POST-PROCESSING, the hypothesis that post-processing helps to improve the summary can be supported. Though all the scores are very close to each other, the summaries WITH POST-PROCESSING summaries achieve better performance.

The precision values obtained indicate that there is a high density of words that are both in the automatic summaries and in the ideal summaries. Still, the summaries WITH POST-PROCESSING have slightly more words in common with the ideal summaries than the summaries WITHOUT POST-PROCESSING, suggesting that the summaries WITH POST-PROCESSING contain more key information than the summaries WITHOUT POST-PROCESSING.

Despite there are probably less in-sequence matches found in the summaries WITH POST-PROCESSING, mainly due to the changes introduced by the post-processing module, these summaries achieve a higher recall value than the summaries WITHOUT POST-PROCESSING. This suggests that the instances that are in the summaries WITH POST-PROCESSING are more relevant than the ones on the summaries WITHOUT POST-PROCESSING. That is, the latter contain less significant information than the former.

Finally, the f-measure score suggests that the summaries WITH POST-PROCESSING include sentences that contain more information present in the ideal summaries.

5.2.6 Conclusions

This Section 5.2 reported the results for the automatic evaluation procedure.

Sentence reduction has been the best performing post-processing module, while in paragraph creation and connective insertion, the differences in the results are so small that may not be considered relevant. Despite this, in both these evaluations, the results

for the summaries built with these modules are slightly better than the ones built without them.

Yet, the results for the overall summarization process seem to be more conclusive. On the one hand, when compared to the random baseline system and to GISTSUMM, SIMBA outperforms both in almost seven percentage points, meaning that its summaries contain much more key information than the other ones. On the other hand, when compared to its own version without the post-processing module, SIMBA also obtains a better performance, though with a small difference. The complexity of evaluating such type of modifications brings to light that automatic metrics are not suited to evaluate readability, cohesion, or textual quality in texts. The post-processing procedure changes the summaries in a way that is hard to detect, so very close automatically obtained scores between both SIMBA's summaries (post-processed and non-post-processed) are not surprising, but also not very much telling.

In summary, a human evaluation is needed to fully assess the improvements that the post-processing procedure can introduce. Still, the automatic evaluation suggests that the strategies used to retrieve the most relevant information and also to remove the redundancy present in the input texts evidence a very competitive performance.

5.3 Human evaluation

The difficulty in resorting to human evaluation has been a considerable hurdle in the development of automatic summarization systems.

There are several problems involved. On the one hand, the selection of the linguistic and textual aspects of the summaries to evaluate is not straightforward. Properties such as cohesion, fluency or readability are hard to define objectively and their assessment differs from person to person depending on one person's background, knowledge or even linguistic skills.

Another problem is the genre from which the summaries were obtained. Depending on the genre, a text can be harder or easier to follow. The same circumstance will apply to its summary. News stories are typically a simple and general type of texts to be read, as they are targeted to a wide audience.

In addition, the size of the input data makes the evaluation even more complex. When evaluating a summary from a single document the information in this document has to

be regarded. But, when verifying the quality of a summary created from a collection of documents, there are several information sources that must be merged in order to find the relevant data present in all of them. Also, the bigger the collection of texts to be summarized, the more complex, arduous, and uncertain the task is.

Finally, the major difficulty to be overcome during a human evaluation is subjectivity. The quality of a summary is subjective. A good summary for an individual might not be a good summary for another individual, whose knowledge about the subject in question or the goal for which the summary is going to be used may not be compared.

Being aware of these challenges, this Chapter reports on the first large-scale human evaluation of multi-document summaries in the Portuguese language.

The remainder of this Chapter is organized as follows. Section 5.3.1 overviews the human evaluation procedure. Section 5.3.2 details the linguistic features targeted in this evaluation. Sections 5.3.3 and 5.3.4 report on both the intrinsic and extrinsic evaluation procedures and their respective results. Finally, Section 5.3.5 discusses the results obtained in the human evaluation procedure.

5.3.1 Overview

The human evaluation procedure was split into two parts. On the one hand, intrinsic evaluation (cf. Section 5.3.3) assessed each post-processing module in itself. On the other hand, the impact of the post-processing procedure on the final summary was also taken into account by performing an extrinsic evaluation (cf. Section 5.3.4).

Intrinsic evaluation is aimed at inferring the quality of a text, not specifically a summary, that was edited using each one of the post-processing methods described in Chapter 4. Extrinsic evaluation, in turn, sought to measure the impact of these modules together, the post-processing procedure, in the context of obtaining a summary from a set of texts.

Both the intrinsic and the extrinsic evaluations were performed using surveys (cf. *Annex F.1*), which were distributed to the users through an e-mail (cf. *Annex F.2*). The email was sent to four groups of users with six people each, obtaining a total of 24 answers. The surveys target the evaluation of the linguistic features discussed in Section 5.3.2. In addition, the opinion of the users about each task and the texts was requested.

5.3.2 Linguistic features

Several types of linguistic dimensions are proposed in the literature. Steinberger and Ježek (2009) proposed four types of aspects that should be analyzed when evaluating text quality: (1) grammaticality, (2) non-redundancy, (3) reference clarity, and (4) coherence and structure. Grammaticality refers to grammar errors. Non-redundancy is related to whether the text contains repetitive information. Reference clarity addresses anaphoric disfluencies related to whether nouns, pronouns and other referential expressions are appropriately used. Coherence and structure have to do with the quality of the summary as a coherent and organized text.

Prior to this study, Over et al. (2007) analyzed the main features used in three evaluation contests: SUMMAC[1], NTCIR[2], and DUC[3]. Concerning linguistic quality, Over et al. (2007) defined three more aspects to evaluate: (1) cohesion, (2) organization, and (3) focus. Cohesion is concerned with sentences fitting as they should with the surrounding sentences. Organization assesses if the content is expressed and arranged in an effective manner. Focus is related to whether that the summary has a focus, and the sentences contain information that is related to the rest of the summary.

Considering these suggestions, and in order to define the linguistic aspects to be evaluated, it is worth recalling the hypotheses stated for this work (cf. Section 1.4).

Hypothesis#1:	*Automatic summarization can be improved by reducing sentences while allowing for the reduction of redundancy and maintaining summary informativeness.*
Hypothesis#2:	*Automatic summarization can be improved by arranging sentences in paragraphs.*
Hypothesis#3:	*Automatic summarization can be improved by inserting discourse connectives.*
Hypothesis#4:	*The automatic post-processing of a summary built by extraction improves the textual quality (cohesion, fluency, readability, etc.) of a summary.*

[1] http://www-nlpir.nist.gov/related_projects/tipster_summac/
[2] http://research.nii.ac.jp/ntcir/index-en.html
[3] http://duc.nist.gov/

On the one hand, the textual quality of the summaries needs to be evaluated. On the other hand, the presence of redundant or repetitive information should also be assessed.

Based also in the goals for this work (described in Section 1.3), the human evaluation was designed to take into account the following aspects:

- Readability
- Cohesion/organization
- Redundancy
- Textual quality

In order to be useful, a summary must be readable. It must be a text whose content is fully understood, requiring no other sources of information.

Cohesion and text organization account for the connectivity between the sentences in the text and the way they are arranged. Paragraph creation and connective insertion procedures aim to address these issues.

Typically, the texts submitted to a multi-document summarizer address the same subject, thus one of the main challenges of this task is the removal of eventual redundant information coming from the input texts. Both the similarity clustering phase of the summarization procedure and the sentence reduction module of the post-processing procedure seek to solve this problem.

Finally, the combination of the summarization procedure with the post-processing module aims to address these issues altogether, by trying to create summaries (1) that contain the key information conveyed by the original set of documents, (2) that do not contain repeated information, and (3) whose textual quality ensures that they are readable and useful. The overall textual quality is then a measure of all these aspects together.

5.3.3 Intrinsic evaluation

The intrinsic evaluation procedure aims to assess the quality of each post-processing module. For each module, a group of six users was asked to rate texts that were created. The texts used in this evaluation were the ideal summaries available on the *CSTNews* corpus. The reason why we used the ideal summaries relies on the fact that those were created by human annotators, being used as the *gold standard summaries* for the automatic

evaluation, which means that these are the best texts available summarizing the original input texts.

As was said before, the intrinsic evaluation was based on surveys. The same strategy was used for all the three modules, sentence reduction, paragraph creation and connective insertion. It relied on a survey created for each of these modules (cf. *Annex F.1*). Each survey contains six tasks. A task is defined by two texts to be evaluated, four questions to be answered and a commentary box. These questions, that target the linguistic features discussed in Section 5.3.2, are stated below.

1. Which text is easier to read and comprehend (Text#1 or Text#2)?
2. Which text is organized in a more effective way (Text#1 or Text#2)?
3. How do you classify the textual quality of Text#1? (0-5)
4. How do you classify the textual quality of Text#2? (0-5)

Each task contains a text resulting from the post-processing module at stake whose quality needs to be evaluated. So six users evaluated six automatically edited texts against their original versions. In each module, 36 opinions were gathered for each type of text, and a total of 108 opinions for all the post-processing modules.

Figures 5.1 and 5.2 show the genre and education of the users that answered to all the surveys for the three post-processing modules. The mean age of this set of users is 38.75 with a standard deviation of 16.75.

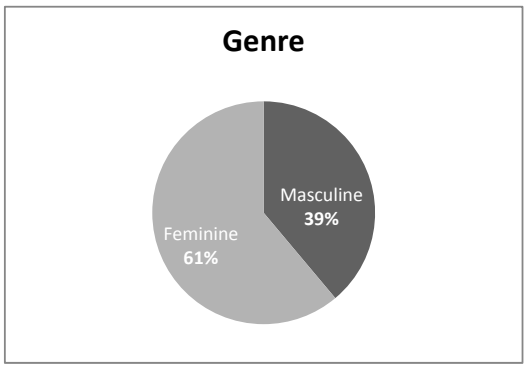


Figure 5.1: User's genre.

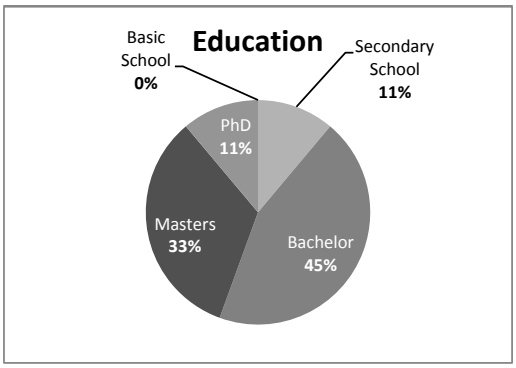


Figure 5.2: User's education.

Results for each of the modules are reported in the following subsections. All the scores reported are an average of the opinions stated by the users over each type of text – edited *vs.* non-edited.

5.3.3.1 Sentence reduction

Two texts are being compared in this evaluation. The first one – NON-REDUCED – is the ideal summary as it is. The second one – REDUCED – is the ideal summary after the sentence reduction procedure described in Section 4.2.1 has been applied. An example of the survey used in this evaluation can be found in *Annex F.1.1*.

Figures 5.3 , 5.4 and 5.5 summarize the answers to the questions stated before.

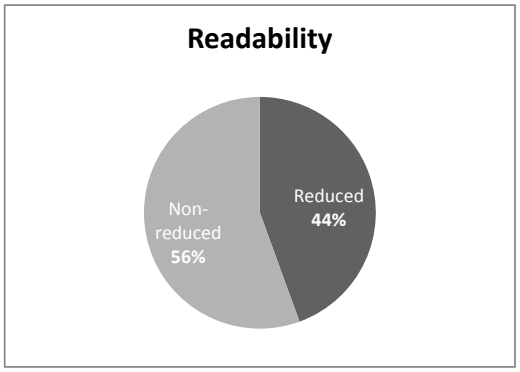


Figure 5.3: Which text is easier to read and comprehend?

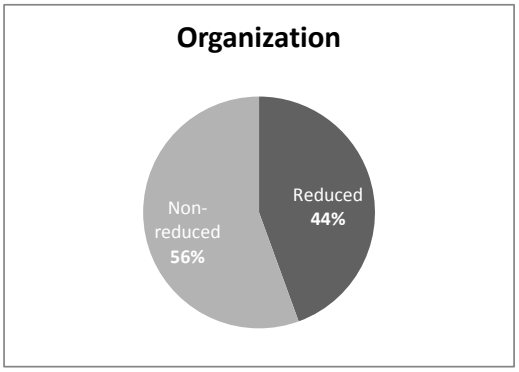


Figure 5.4: Which text is organized in a more effective way?

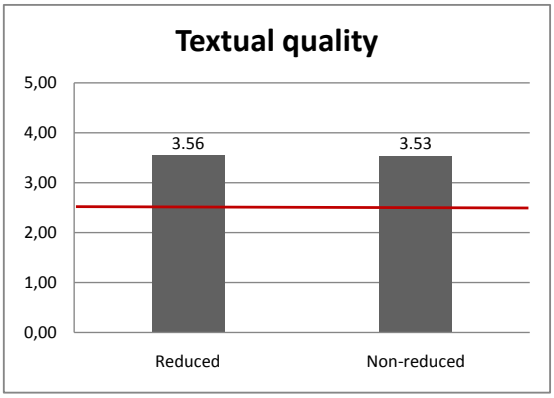


Figure 5.5: How do you classify the textual quality of both texts? (0-5)

When asked which text was easier to read and comprehend, users stated that the NON-REDUCED are easier to read and comprehend. Then, when asked which text was better organized, users also pointed the NON-REDUCED texts as being better organized. Finally, a quantitative metric (0-5) for the overall textual quality of the texts was obtained. The

users rated the REDUCED texts with 3.55 and the NON-REDUCED with 3.52. Despite they have considered that the NON-REDUCED texts are easier to understand and better organized, they stated still that the REDUCED ones had a better overall textual quality. Finally, in the commentary box, some users reported that both texts were very similar, and that it was hard to distinguish between them. Also, they informed that in general the REDUCED text was more summarized than the NON-REDUCED one.

Despite the REDUCED texts having less information than the NON-REDUCED ones, users considered that the REDUCED texts were better in terms of textual quality, though the readability and organization of the REDUCED texts might have been affected by the sentence reduction procedure. In general, we can conclude that users considered both texts very similar and that the sentence reduction procedure does not remove too many information from the texts as they still convey the same information as the NON-REDUCED ones.

5.3.3.2 Paragraph creation

Again two texts are being compared in this evaluation. The first one – WITHOUT PARAGRAPHS – is the ideal summary as it is. The second one – WITH PARAGRAPHS – is the ideal summary after the paragraph creation procedure described in Section 4.3.1 has been applied. An example of the survey used in this evaluation can be found in *Annex F.1.2*.

Figures 5.6 , 5.7 and 5.8 summarize the answers to the questions stated before.

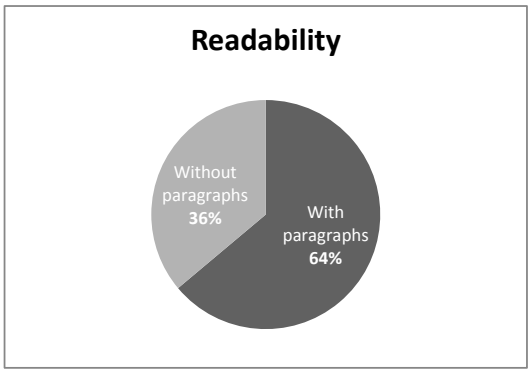


Figure 5.6: Which text is easier to read and comprehend?

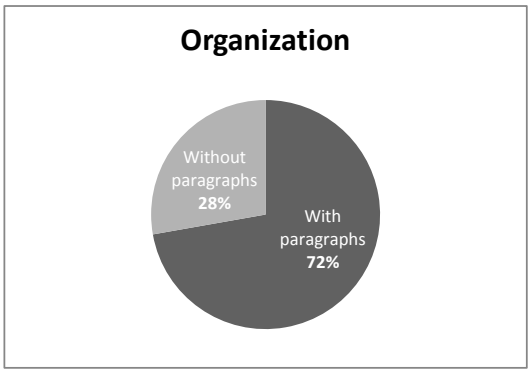


Figure 5.7: Which text is organized in a more effective way?

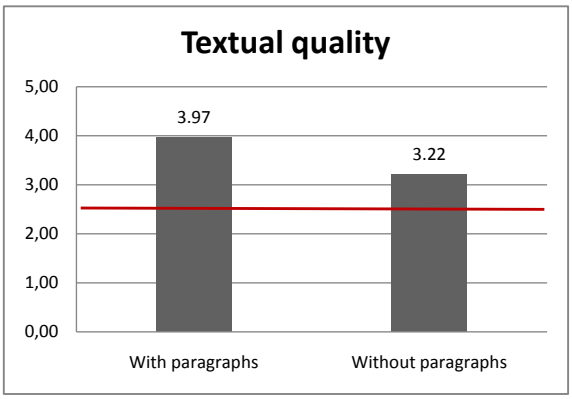


Figure 5.8: How do you classify the textual quality of both texts? (0-5)

When asked which text was easier to read and comprehend, the majority of the opinions (64%) determined that the texts WITH PARAGRAPHS are easier to read an comprehend. Moreover, users considered that the texts WITH PARAGRAPHS were better organized than the texts WITHOUT PARAGRAPHS, also by a large majority 72%. Finally, a quantitative metric (0-5) for the overall textual quality of the texts was obtained. The users rated the text WITH PARAGRAPHS with 3.92 and the text WITHOUT PARAGRAPHS with 3.22. Despite both texts have achieved a positive grade, a huge difference has been found by the users between these texts, as the texts WITH PARAGRAPHS have been considered much better than the texts WITHOUT PARAGRAPHS. In the commentary box, users reported that the texts WITH PARAGRAPHS were easier to follow and that these had a better structure.

After this evaluation procedure, we can conclude that the texts WITH PARAGRAPHS overcome the texts WITHOUT PARAGRAPHS in all the metrics considered. Texts WITH PARAGRAPHS are thus better regarding textual quality, readability and organization.

5.3.3.3 Connective insertion

Two texts are being compared in this evaluation. The first one – WITHOUT CONNECTIVES – is the ideal summary as it is. The second one – WITH CONNECTIVES – is the ideal summary after the paragraph creation procedure and the discourse connective insertion described in Section 4.3.2 has been applied. An example of the survey used in this evaluation can be found in *Annex F.1.3*.

Figures 5.9 , 5.10 and 5.11 summarize the answers to the questions stated before.

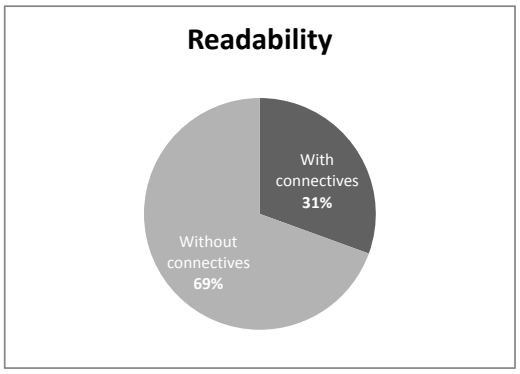


Figure 5.9: Which text is easier to read and comprehend?

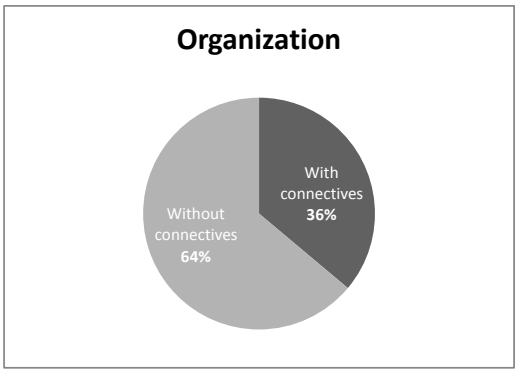


Figure 5.10: Which text is organized in a more effective way?

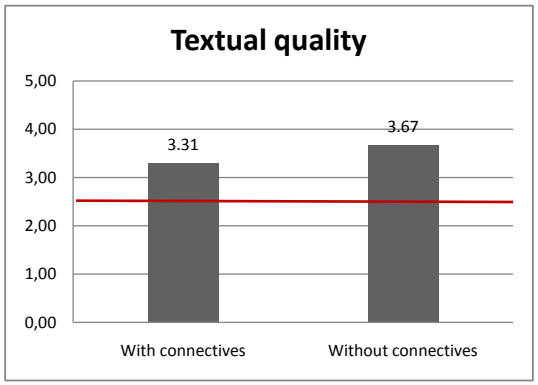


Figure 5.11: How do you classify the textual quality of both texts? (0-5)

When asked which text was easier to read and comprehend, users stated that the texts WITHOUT CONNECTIVES were significantly easier to understand (69%). Then, when asked which text was better organized, users also stated that the texts WITHOUT CONNECTIVES were better organized, though in a slightly lower percentage (64%).

Finally, a quantitative metric for the overall quality of the texts was obtained. From their opinions, we can conclude that texts WITHOUT CONNECTIVES are better than the ones WITH CONNECTIVES, despite the difference being smaller than the one in the other questions. This means that, though users found the texts WITHOUT CONNECTIVES better, texts WITH CONNECTIVES still have a positive grade.

After this evaluation procedure, we can conclude that the texts WITHOUT CONNECTIVES have been considered better than the texts WITH CONNECTIVES, regarding textual quality, readability and organization.

5.3.4 Extrinsic evaluation

The extrinsic evaluation procedure aims to assess the quality of the post-processing procedure as a whole. A group of six users was asked to rate the quality of two automatic summaries not only regarding them as texts, but also as summaries built from a set of input texts.

The same corpus used in the automatic evaluation, *CSTNews* (cf. Section 5.1), was used in this evaluation. Recall that this corpus is composed by 50 sets of documents. For this evaluation, three sets of this corpus were used. For each set, two types of summaries were created. One of them was first summarized and then post-processed – WITH POST-PROCESSING –, and the other one was only summarized – WITHOUT POST-PROCESSING.

This extrinsic evaluation procedure was based on a survey. An example of the survey can be found in *Annex F1.4*. The survey contained three tasks. A task is defined by three input texts, two summaries of them to be evaluated, and a questionnaire. The questionnaire, that targets the linguistic features discussed in Section 5.3.2, contains two parts and ends with a commentary box.

The first part defines the questions (shown below) about the summaries when taking into account the input texts.

1. Which text is the best summary for the input texts (Summary#1 or Summary#2)?
2. Considering the input texts, how much relevant information each summary contains? (0-5)
3. How much repeated information each summary contains? (0-5)

The second part includes the questions (shown below) about the summaries as texts.

1. Which text is easier to read and comprehend (Text#1 or Text#2)?
2. Which text is organized in a more effective way (Text#1 or Text#2)?
3. How do you classify the textual quality of Text#1? (0-5)
4. How do you classify the textual quality of Text#2? (0-5)

In this evaluation, six users evaluated six automatically created texts, providing 36 opinions about the automatically obtained texts (WITH and WITHOUT POST-PROCESSING).

Figures 5.12 and 5.13 show the genre and education of the users that answered this survey. The mean age of this set of users is 38.17 with a standard deviation of 13.56.

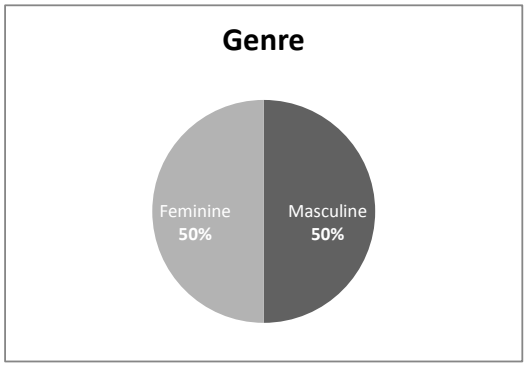


Figure 5.12: User's genre.

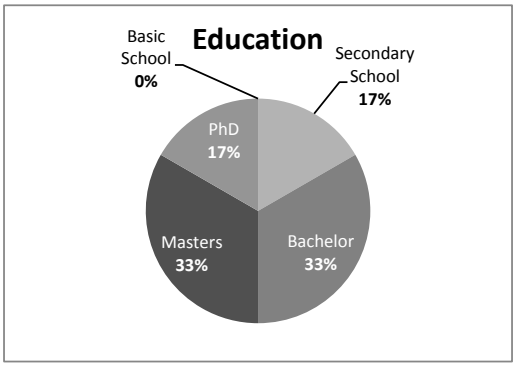


Figure 5.13: User's education.

Results for this evaluation are reported in the following subsection. All the values reported are an average of the opinions stated by the users over each type of text.

As was said above, two texts are being compared in this evaluation. The first one – WITHOUT POST-PROCESSING – is a summary created by running the summarization procedure described in Chapter 3. The second one – WITH POST-PROCESSING – is a summary created by SIMBA, which means that it was built by running both the summarization algorithm (cf. Chapter 3) and the post-processing module (cf. Chapter 4).

Figures 5.14 , 5.15 and 5.16 summarize the answers to the questions for the first part of the survey, where users were asked about the summaries.

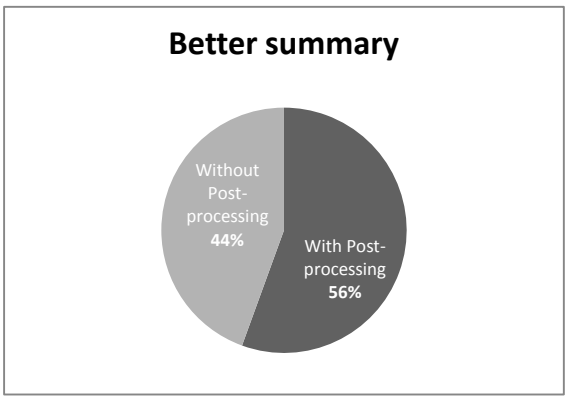


Figure 5.14: Which text is the best summary for the input texts?

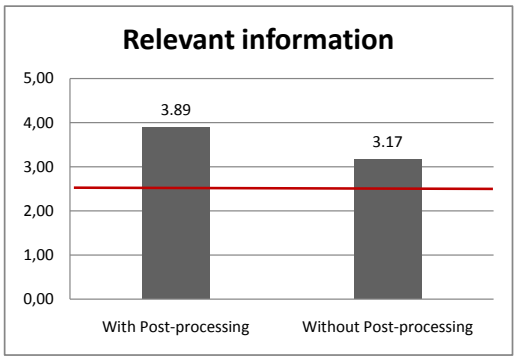


Figure 5.15: Considering the input texts, how much relevant information each summary contains? (0-5)

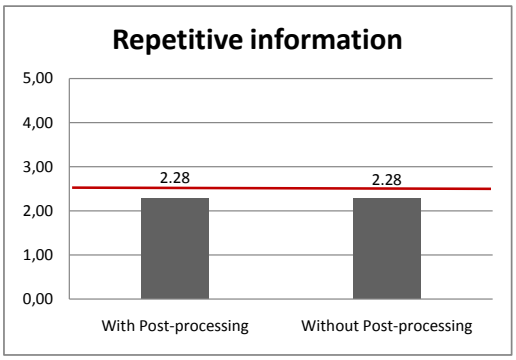


Figure 5.16: How much repeated information each summary contains? (0-5)

When asked about which would be the best summary for the input texts, the user's opinion is that the summaries WITH POST-PROCESSING are better than the ones WITHOUT POST-PROCESSING. Moreover, users stated that, considering the information present in the input texts, the summaries WITH POST-PROCESSING have more relevant information than the summaries WITHOUT POST-PROCESSING. Finally, they considered that both types of summaries have few repeated information.

Afterwards, users were asked about the quality of the summaries regarding them as texts. Figures 5.17 , 5.18 and 5.19 summarize the answers to the questions for the second part of the survey.

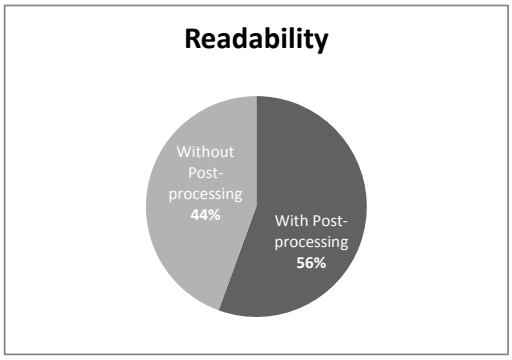


Figure 5.17: Which text is easier to read and comprehend?

Figure 5.18: Which text is organized in a more effective way?

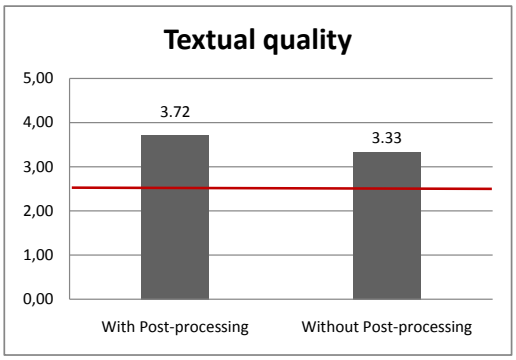


Figure 5.19: How do you classify the textual quality of both texts? (0-5)

When asked which text was easier to read and comprehend, users stated that the summaries WITH POST-PROCESSING were the best ones. When asked which text was better organized, users were unable to decide between both summaries, as the opinions are evenly divided. Moreover, a quantitative metric (0-5) for the overall textual quality of the summaries was obtained. The users rated the summaries WITH POST-PROCESSING with 3.72 and the summaries WITHOUT POST-PROCESSING with 3.33. Despite both summaries achieved a positive grade, the users found them different, as the summaries WITH POST-PROCESSING have been considered better than the summaries WITHOUT POST-PROCESSING. In the commentary box, users reported that the texts WITH POST-PROCESSING were easier to follow and that their topics were better organized. They stated that in general all the summaries were very good. The users finally mentioned that the summarizer was able to include the most relevant information from the input texts and to ignore the redundant one.

After this evaluation procedure, we can conclude that both the summaries and the texts WITH POST-PROCESSING were pointed as being better than the texts and the summaries WITHOUT POST-PROCESSING. When considering the summaries as texts, the ones WITH POST-PROCESSING were stated as easier to read and with better textual quality than the ones WITHOUT POST-PROCESSING. When considering the texts as summaries from the input texts, also the texts WITH POST-PROCESSING were indicated as being better summaries and with more relevant information than the ones WITHOUT POST-PROCESSING.

5.3.5 Conclusions

Human evaluation included two types of approaches: an intrinsic and an extrinsic. While the intrinsic one aimed at evaluating each post-processing step alone, the extrinsic one

sought to evaluate the complete summarization procedure, including the post-processing module.

In the evaluations reported in the previous subsections, we concluded that:

- The users considered that the REDUCED texts were very similar to the NON-REDUCED ones despite having less information than these.
- The users clearly preferred the texts WITH PARAGRAPHS over the ones WITHOUT PARAGRAPHS.
- The users found the texts WITH CONNECTIVES worse than the texts WITHOUT CONNECTIVES.
- The users reckoned that the texts WITH POST-PROCESSING were better than the texts WITHOUT POST-PROCESSING.
- The users concluded that the texts WITH POST-PROCESSING were better summaries of the input texts.

There are also some further interesting points to note. In the intrinsic evaluation for the sentence reduction module, users stated that the REDUCED texts had a better overall textual quality. When extrinsically evaluating SIMBA, users found that the summaries WITH POST-PROCESSING had the same amount of repeated information as the summaries WITHOUT POST-PROCESSING. Moreover, according to the user's opinions, the summaries WITH POST-PROCESSING convey more relevant information than the ones WITHOUT POST-PROCESSING. This suggests that the sentence reduction procedure, along with the clustering by similarity procedure, helps to reduce the redundancy in the automatic summaries, while helping to improve the informativeness of the summary.

In addition, users also stated that the paragraph creation module was very helpful when reading and understanding the text. Also, the readability of the summaries WITH POST-PROCESSING has been pointed out as better than the one of the summaries WITHOUT POST-PROCESSING. This means that the paragraph creation module helps to improve the readability of the summary and thus the overall automatic summarization procedure.

Finally, even though the intrinsic evaluation of the connective insertion procedure has reported a poor performance, it does not impact in the extrinsic evaluation of SIMBA, where summaries WITH POST-PROCESSING were rated as being good, and better than the WITHOUT POST-PROCESSING ones.

5.4 Discussion

The evaluation performed in this work had two main goals. On the one hand, automatic evaluation aims to determine the quality of the statistical procedure that selects and ranks the sentences to be part of the final summary by means of commonly used automatic metrics. On the other hand, human evaluation analyzes the textual quality of the summaries, that is, it seeks to validate whether the post-processing procedure can improve the quality of the summaries.

Automatic evaluation reports good results. Considering the baselines for each post-processing module, the summaries created by the latter achieved better performance than the ones without them. Recall that sentence reduction has been the best performing post-processing module, overcoming the baseline in almost three percentage points. On the contrary, for both paragraph creation and connective insertion the differences between the summaries with and without these modules are very small. So, no firm conclusions could be drawn from their automatic evaluation. In spite of this, the summaries built with these modules have achieved a better performance than the ones without them.

Yet, the human evaluation brought more discrimination. Users found the REDUCED summaries very similar to the NON-REDUCED ones, still they considered the REDUCED summaries better texts. Also, sentence reduction was the best performing module in the automatic evaluation, suggesting that the REDUCED summaries contain more relevant information than the NON-REDUCED summaries. Thus, we can conclude that sentence reduction not only does not remove relevant information from the texts, as it also makes room for more important information to be included in the summary, producing better summaries.

Recall that the automatic evaluation of both paragraph creation and connective insertion reported insignificant differences between the summaries created with and without these modules. Still, in the human evaluation procedure, they were rated very differently and in a very conclusive way. Summaries WITH PARAGRAPHS were considered much better than summaries WITHOUT PARAGRAPHS. In the opposite direction, summaries WITH CONNECTIVES were considered worse than summaries WITHOUT CONNECTIVES. This is a very interesting result, as we can see that human evaluation can help to perceive the subtle differences in the texts that automatic evaluation was not able to identify.

Regarding the automatic evaluation, SIMBA's summaries, representing the complete summarization procedure plus the post-processing process – WITH POST-PROCESSING –, not only overcame all the baselines, but also the summaries WITHOUT POST-PROCESSING. This result has mainly been achieved by combining the statistical methods that select the sentences to be part of the summary, with the sentence reduction procedure, which removes redundant information, and the compression review process that seeks to insert more significant information in the summary.

This fact has been confirmed in the human evaluation, where users stated that the summaries WITH POST-PROCESSING have more relevant information than the summaries WITHOUT it. Moreover, when evaluating these summaries against their input texts, users also preferred the summaries WITH POST-PROCESSING over the summaries WITHOUT POST-PROCESSING, stating that the summaries WITH POST-PROCESSING (1) were better summaries for the input texts, (2) contained more relevant information, and (3) had better textual quality. Interestingly, the poor performance of the connective insertion procedure had less impact in the overall textual quality of the summaries. In fact, the better performance of both sentence reduction and paragraph creation helped to enhance the overall quality of the summaries WITH POST-PROCESSING.

Our work contributes with the first attempt to combine statistical methods with discourse structure in automatic generated summaries. Two lessons have been learned in the complete evaluation process. On the one hand, statistical methods enable the selection of relevant content to be part of the summary. On the other hand, the post-processing modules in general help to enhance summary textual quality, though the approach that uses discourse structure and connectives to enhance summaries can be improved.

Our work provides empirically confirmed support to all our working hypotheses, except Hypothesis#3.

5.5 Summary

This Chapter presented the evaluation carried out to determine the quality of the methods proposed in Chapters 3 and 4.

Automatic evaluation was performed to mainly assess the informativeness of the summaries generated automatically. The summaries built by all the modules proposed in our

work were compared with summaries built by automatic baselines. Results for all these comparisons were presented and discussed.

Human evaluation was used to determine the quality of the summaries. Both intrinsic and extrinsic processes were carried out, in order to infer the quality not only of the modules by themselves – intrinsic –, but also of the complete procedure (summarization together with post-processing) – extrinsic. Results for both these evaluation procedures were also presented and discussed. They provide empirically confirmed support to all our working hypotheses, except Hypothesis#3.

CHAPTER 6

CONCLUSIONS

The conclusions of this dissertation are presented in this Chapter. First, the major contributions are reported (Section 6.1). Afterwards, taking into account these contributions and the results expressed in this dissertation, future directions of research are considered (Section 6.2). This dissertation is then concluded with some final remarks (Section 6.3).

6.1 Contributions

A text built with sentences extracted from different sources is expected to form a fragile whole, whose elements may not be fully textually connected. The work reported here defines the very first approach to the improvement of the textual quality of an automatically generated summary. Locating relevant information using simple and yet effective and efficient algorithms, while focusing on relating and organizing the gathered information to form a consistent whole are the main goals of this work.

Along with theoretical contributions, this work has produced a practical outcome, that is an automatic multi-document summarization system[1] for the Portuguese language. This system performs extractive summarization, so the summaries produced are

[1] A web version of SIMBA can be found in `http://lxsimba.di.fc.ul.pt/`. The code for this system can be found in `https://github.com/sarabsilveira/simba`.

composed by sentences retrieved from the input texts. By means of two clustering phases executed in sequence, more redundant sentences are discarded (clustering by similarity) and more relevant sentences are identified (clustering by keywords), based on commonly used scoring metrics (*tf-idf*). After the sentences for the summary have been selected, a post-processing module is applied in order to enhance the textual quality of the text.

The post-processing module is a pioneer approach to enhance summaries produced by extraction methods. Within this module there are three innovative methods that together seek to improve the quality of automatically generated summaries.

Sentence reduction seeks to create simpler and more concise elements to be used in the summary. A rule-based approach was carried out based on linguistic knowledge applied over the result of a constituency parser. The structures that may have less essential information are firstly identified and then removed if they deteriorate the sentence score.

Paragraph creation intends to define a global structure for the text, by grouping the sentences placed in the same keyword cluster.

Discourse connective insertion strives to relate adjacent sentences, within a paragraph, by taking into account the discourse relations they share. In this context, the very first discourse corpus for the Portuguese language has been built. Also, the first approach to identify discourse relations between sentences in Portuguese has been carried out. The same way, an exploratory study on inserting discourse connectives between two unseen sentences has been performed. It is relevant to mention that this is the topic for *CoNLL Shared Task for 2015*[2], despite it being restricted to systems that work with English.

The combination of these three modules – sentence reduction, paragraph creation and connective insertion – constitutes an innovative approach to address the problem of lack of cohesion on automatically generated summaries, specially when built through extraction procedures.

Finally, a comparative evaluation between the existing systems that perform automatic summarization in Portuguese has been carried out, to study whether this approach produces improved results. Moreover, a human evaluation has also been accomplished to assess the textual quality of the summaries produced.

The post-processing procedure is the key differentiating factor of this work, as reflected in its hypotheses which seek to address the impact of the post-processing procedure over automatic summarization. Let us now recall these hypotheses.

[2] `http://www.cs.brandeis.edu/~clp/conll15st/`

Hypothesis#1:	*Automatic summarization can be improved by reducing sentences while allowing for the reduction of redundancy and maintaining summary informativeness.*
Hypothesis#2:	*Automatic summarization can be improved by arranging sentences in paragraphs.*
Hypothesis#3:	*Automatic summarization can be improved by inserting discourse connectives.*
Hypothesis#4:	*The automatic post-processing of a summary built by extraction improves the textual quality (cohesion, fluency, readability, etc.) of a summary.*

When considering the conclusions drawn in both evaluation procedures performed – automatic (cf. Section 5.2) and human (cf. Section 5.3) –, three out of four hypotheses received empirical confirmation.

On the one hand, the automatic evaluation procedure reports a good performance of this system when compared to other available systems and to the baselines constructed for the purpose of this evaluation. In what concerns the post-processing modules, sentence reduction was the best performing module, supporting the validation of Hypothesis#1. Yet, the results for paragraph creation and connective insertion tasks were inconclusive at this point, as the modifications performed are so subtle that automatic metrics were not granular enough to perceive them. Still, these results reported a slightly better performance of the summaries with paragraphs and with connectives.

Additionally, human evaluation brought some more clarification about the post-processing modules. The summaries built with the reduction module were considered content-wise very similar to the summaries without this module, meaning that, despite having less words, these can basically convey the same key information. From this, we can conclude that Hypothesis#1 has been successfully validated, as reduced summaries contain less redundant information, though maintain the summary informativeness.

Also the summaries with paragraphs achieved a high performance, being considered much better than the ones without them, confirming this way Hypothesis#2.

In a different way, summaries with connectives were considered worse than the ones without them, not permitting to validate Hypothesis#3.

Despite this, and following on the good results obtained by both the sentence reduction and the paragraph creation procedures, the post-processed texts were considered better than the non-post-processing ones. Additionally, these texts were considered better summaries for the input texts. Both these two facts determine that also Hypothesis#4 has been empirically validated.

As these conclusions point out, it can be considered that the major goals of this work have been accomplished. SIMBA produces summaries with a an improved textual quality, built through an automatic system. In addition, SIMBA contributes positively to improve the state of the art by incorporating, in an automatic summarization system, a summary post-processing module. Finally, a study on the impact of post-processing summarized texts has been performed from which we concluded that the post-processing of summaries can improve their textual quality.

6.2 Future work

There are several open challenges that can be addressed in the future in order to further pursue the research lines explored here.

Regarding the **summarization process** as a whole, the score values assigned to the sentences during both clustering processes (similarity and keywords) can be optimized to enhance sentence selection.

Still, other types of scores can be created: (i) cluster size and (ii) document scores. The scores based on cluster sizes can take into account the size of each one of the clusters or both clusters (similarity and keywords). The scores of the sentences within those clusters can also be helpful to better distinguish the most relevant information within the collection of texts. Document scores are related to the original documents. For instance, a document score can be defined by the average scores of its sentences, so the sentences from documents with higher scores are more likely to be included in the summary.

Furthermore, a tool that analyzes time events in texts written in Portuguese (Costa, 2013) can be used to improve the ordering procedure. By identifying the time expressions in each sentence, it would be possible to order the sentences based on these, performing a more reliable order, when considering the sequence of the events mentioned in the texts.

Considering the **post-processing procedure**, there may be several ways to explore to ameliorate this procedure. As users pointed out in the human evaluation procedure, con-

nective insertion is by far the task where most of the refinements should be made. While building the discourse corpus in order to accomplish this specific task, a closed list of connectives was built. Eventually by restraining this list to one connective for each class, more granular features can be selected. A single connective would have a very specific construction and this construction could help to refine the feature set and to create more accurate classifiers, resulting in a better classification.

Moreover, recall from Section 4.3.2.5, that not all the available data was exploited, as only one part of the corpus was used as a training corpus. Extending the training datasets would be important, in order to enhance the accuracy of the classifiers. This proved to be effective in the first classifier build – *Null vs. Relations* – as we were able to improve the accuracy of the classifier in almost seven percentage points simply by extending the training dataset. So, extending the training datasets for all the classifiers created for the *Relations vs. Relations* phase could also improve the classification.

Likewise, the paragraph creation procedure can be expanded, though good results were already obtained for this task. For instance, a sentence distance metric can be used to identify the sentences that should be put together. In fact, the decisions concerning the definition of the paragraphs are closely linked to the summarization ordering procedure, which can also be upgraded. For instance, the grouping and ordering strategies could benefit from the discourse relations found between the sentences. A structure for the final summaries could be previously defined considering the typical discourse structure of a text. So, these relationships would be used to define the order of the sentences by grouping them according to that structure. This way, discourse structure would also account for the text as a whole, instead of taking into account only the sentence-level.

Finally, a note to other possible post-processing tasks. A well known problem in creating summaries from multiple sources is the dangling pronouns. This is an issue that has not been addressed in this post-processing module. Indeed, a reference adjustment task would be very interesting to include in this procedure. By removing phrases within a sentence, the sentence reduction process can produce missed references. Moreover, the sentences selected in the content selection phase could also contain references to prior information. Reference adjustment seeks to locate missed references and correct them. In addition, subjects can be repeated in subsequent sentences. Besides, the subject references may vary and, for instance, the same person may be named differently in different summary sentences. These repetitions can be removed and replaced by null

subjects or pronouns. Joining references to the same subject or replacing those references would also smooth the text. Anaphora resolution has been a matter of constant study in Natural Language Processing, and it could be of a great help to include it as a post-processing task. Also, a study about co-reference cohesion in extraction-based summaries (Gonçalves, 2008) has concluded that correcting missed references would improve the informativeness of the summaries. Not only resolving anaphoric expressions, but also replacing references to the same subject, could improve the quality of the summary.

In summary, as this is the first approach to a statistical post-processing process, the main decision was to build the simplest algorithm, in order to draw some conclusions about the improvements that such a process would bring to a text, so that, in the future, other possibilities could be pursued and experimented with. This post-processing module is a baseline system, still open for improvements, not only in its already developed stages, but also in new stages considered relevant to be included.

6.3 Final remarks

This dissertation reports on the investigation of the inclusion of post-processing tasks in multi-document summarization in order to enhance the textual quality of the summary. When considering both automatic and human evaluation procedures, the approach pursued has been validated in general, as the summaries produced have been considered better than the other texts under comparison.

Regarding a subjective evaluation of the textual quality of the summaries, performed by users, not only the post-processed summaries have been considered better texts than the ones without post-processing, but also the summaries with post-processing were considered better summaries for the input texts. Taking into account that the approach presented is a very simple one, there is a strong indication that a refined post-processing module, including all the tasks discussed in the previous section, can be of great help when improving automatically generated summaries.

Still, this work reports on the very first approach to multi-document summarization for the Portuguese language focused on the textual quality of the final texts, which shall be consumed by human users. This approach combined with post-processing tasks creates summaries that proved to be more readable, and thus can be useful in user's everyday tasks.

ANNEXES

A Original texts

A.1 BNDES destina US$ 1 bi para pequenas empresas

Empréstimo deve ser pedido nos bancos; garantias são de 130% do valor – CARMEN BARCELLOS

O BNDES (Banco Nacional de Desenvolvimento Econômico e Social) tem disponível US$ 1 bilhão em cinco linhas de financiamento para pequenas empresas industriais, comerciais e de serviços.

No final de junho, foi lançada a linha Enter/BNDES, aberta também a profissionais liberais.

Destina-se à compra de equipamentos de computação e programas nacionais e cobre gastos com a implantação de projetos de informatização e treinamento.

O crédito chega a 85% do total do investimento, com juros de 10% ao ano (veja quadro ao lado).

O BNDES está acertando convênios com entidades de classe, que determinarão produtos e serviços a ser financiados.

A linha Finame automático é exclusiva para a compra de máquinas e equipamentos. Financia de 80% a 90% do valor dos produtos, conforme a região em que a empresa se localiza.

O financiamento só vale para produtos novos, nacionais e de fabricantes cadastrados pelo Finame.

O POC automático financia investimentos em instalações, infra-estrutura, programas de treinamento, qualidade e produtividade, controle do meio ambiente e desenvolvimento tecnológico. O limite do crédito é de US$ 1 milhão.

Indústrias podem solicitar até 30% do total financiado para ser utilizado como capital de giro.

O BNDES automático segue as mesmas regras do POC automático, porém para financiamentos entre US$ 1 milhão e US$ 3 milhões.

Também há recursos para a importação de equipamentos, através do POC/Importação. O crédito é limitado a US$ 1 milhão.

"O BNDES está destinando US$ 1 bilhão este ano para financiamentos a micro e pequenas empresas", afirma Laércio Gonçalves, 47, chefe do escritório do BNDES em São Paulo.

Para ter acesso aos recursos, o empresário deve apresentar no banco de sua preferência um orçamento do produto ou projeto que pretende implantar na empresa.

O banco avalia a proposta e solicita outros documentos referentes à empresa. A única exceção é o Enter/BNDES, em que se deve procurar antes uma entidade de classe.

Gonçalves afirma que os prazos para liberação dos recursos variam conforme o tempo que cada banco leva para analisar as propostas e enviá-las para aprovação do BNDES.

Na maioria dos financiamentos, o empresário recebe o valor total do crédito.

Apenas no POC e no BNDES automático a liberação ocorre em parcelas, de acordo com o plano de execução do projeto.

As garantias ficam a critério de cada banco. Em geral, giram em torno de 130% do financiamento. No Finame, pode ser o próprio equipamento.

A.2 Julia Roberts named "world's most beautiful" by People.

Roberts, 42, the mother of 5-year-old twins and a 2 year-old son, is joined by Halle Berry, Angelina Jolie and Jennifer Lopez on the 2010 list.
Garry Marshall, who directed Roberts in her breakout role as a prostitute with a heart of gold in the movie "Pretty Woman" 20 years ago, said she was "as beautiful as ever."
She's much more centered and calm now and not so stressed. She is quite the mom – very nurturing," Marshall told People.
Roberts, who won an Oscar in 2001 for her lead role in "Erin Brockovich", will be seen next in August in the movie "Eat, Pray, Love" based on the best-selling book of the same name by Elizabeth Gilbert.
Ashley Greene, 23, who plays a vampire in the "Twilight" hit movie franchise, was among the newcomers to the widely-read list that annually is a measure of celebrity status.
Her "Twilight" co-star, heartthrob Robert Pattinson, as well as Canadian teen singer Justin Bieber and 2009 "American Idol" runner-up Adam Lambert made it to the male section of "world's most beautiful", while actresses Jessica Biel, Jessica Alba and singer Jessica Simpson were featured in a new "Beautiful Jessicas" section.
While Roberts appeared on the cover of the People magazine special edition, other celebrities were not ranked but included Jennifer Aniston, Rihanna, Taylor Swift, Beyonce, Bradley Cooper, Johnny Depp and Patrick Dempsey.
Last year's title was awarded by People magazine to actress Christina Applegate, who had previously made public her battle with breast cancer and double mastectomy.

B Supporting tools

This annex presents the tools, developed in NLX group, that have been used in the development of the automatic summarization system. A website – LX-Center (Branco et al., 2009) – was developed to group all these tools in their webservices version. The descriptions presented in this annex can be obtained by visiting LX-Center[3].

B.1 LX-Suite

LX-Suite (Branco and Silva, 2006) is a set of shallow processing tools for Portuguese, with state of the art performance. It comprises a pipeline of six modules, namely a sentence chunker, a tokenizer, a POS tagger, and a nominal and a verbal featurizer and lemmatizer. LX-Suite also includes a web service – LX-Service (Branco et al., 2008) – that allows the remote execution of the first three modules of the pipeline.

[3] http://lxcenter.di.fc.ul.pt

LX-Chunker

LX-Chunker is the LX-Suite sentence chunker, which marks sentence boundaries with <s> ... <s> and paragraph boundaries with <p> ... <p>. It also unwraps sentences split over different lines. An f-score of 99.94% was obtained when testing on a 12,000 sentence corpus accurately hand tagged with respect to sentence and paragraph boundaries.

LX-Tokenizer

LX-Tokenizer performs several tasks: (1) segments text into lexically relevant tokens, using whitespace as the separator[4] (*um exemplo* → |*um*|*exemplo*|); (2) expands contractions[5] (*do* → |*de*_|*o*|); (3) marks spacing around punctuation or symbols[6] (*8 . 6* → |*8*|*.*/|*6*|); (4) detaches clitic pronouns from the verb(*afirmar-se-ia* → |*afirmar-CL-ia*|*-se*|); (5) handles ambiguous strings. These are words that, depending on their particular occurrence, can be tokenized in different ways (*deste* → |*deste*| – *verb.* or *deste* → |*de*|*este*| – *contraction*). This tool achieved an f-score of 99.72%.

LX-Tagger

LX-Tagger assigns a single morpho-syntactic tag, from a predefined tagset, to every token. The tag is attached to the token, using a / (slash) symbol as separator – for instance, *um exemplo* → *um/IA exemplo/CN*. This tagger was developed with TnT (Brants, 2000) software over 90% of a small, 260k token, accurately hand tagged corpus. An accuracy of 96.87% was obtained for this tool.

LX-Featurizer (nominal)

LX-Featurizer performs two tasks. First, it assigns inflection feature values to words from the nominal categories. Namely, Gender (masculine or feminine), Number (singular or plural) and, when applicable, Person (1st, 2nd and 3rd): *os/DA gatos/CN* → *os/DA#mp gatos/CN#mp*. Afterwards, it assigns degree feature values (diminutive, superlative and comparative) to words from the nominal categories: *os/DA gatinhos/CN* → *os/DA#mp gatinhos/CN#mp-dim*. This tool has 91.07% f-score.

[4] In these examples, the | (vertical bar) symbol is used to mark the token boundaries more clearly.
[5] The first element of an expanded contraction is marked with an _ (underscore) symbol.
[6] The * and the */ symbols indicate a space to the left and a space to the right, respectively.

LX-Lemmatizer

LX-Lemmatizer assigns a lemma to words. This lemma corresponds to the form that one would find in a dictionary, typically the masculine singular form. The lemma is inserted into the token, with "/" (slash) as a delimiter – *gatas/CN#fp → gatas/GATO/CN#fp*. This tool has 97.67% f-score.

LX-Lemmatizer and Featurizer (verbal)

LX-Lemmatizer and Featurizer (verbal) assigns a lemma and inflection feature values to verbs. The lemma corresponds to the infinitive form of the verb. The lemma is inserted into the token, with / (slash) as a delimiter – for example, *escrevi/V → escrevi/ESCREVER/V#ppi-1s*. The tool disambiguates among the various lemma-inflection pairs that can be assigned to a verb form, achieving 95.96% accuracy.

B.2 LX-Parser

LX-Parser (Silva et al., 2010) is a probabilistic constituency parser for Portuguese, which performs a syntactic analysis of Portuguese sentences in terms of their constituency structure, through a probabilistic grammar. It was trained over an annotated corpus of Portuguese sentences (Branco et al., 2010), and can be used as a stand-alone tool that takes a sentence as input and retrieves its constituency tree. LX-Parser is supported by the Stanford Parser (Klein and Manning, 2003), which is a statistical parser that is trained over a previously annotated corpus. Under the Parseval metric it achieved an accuracy of 88%.

B.3 LX-NER

LX-NER (Ferreira et al., 2007) is a named entity recognizer (NER) for Portuguese. It handles two categories of expressions: number-based expressions and name-based expressions. Number-based expressions include numbers, measures (currency, time, scientific units), time (date, time periods, time of the day), and addresses (global subsection, local subsection and zip code). Name-based expressions include names of persons, organizations, locations, events, works and miscellaneous.

LX-NER number-based module scored 85.19% precision and 85.91% recall, while the name-based module scored 86.53% precision and 84.94% recall.

C Summarization example

C.1 Translation

This Section shows the translation of the texts used during the description of the summarization procedure. The numbers in brackets are not part of the text. Those represent the sentence absolute positions for further reference. The paragraphs in the original texts have been maintained. These texts have been translated by hand, keeping, whenever possible, the alignment with the Portuguese version.

Text #1 – retrieved from the newspaper *Estadão*

(1) A crash in the town of Bukavu in the eastern Democratic Republic of Congo (DRC), killed 17 people on Thursday afternoon, said on Friday a spokesman for the United Nations.

(2) The victims of the accident were 14 passengers and three crew members. (3) All died when the plane, hampered by bad weather, failed to reach the runway and crashed in a forest 15 kilometers from Bukavu airport. (4) According to airport sources, the crew members were of Russian nationality.

(5) The plane exploded and caught fire, said the spokesman of the UN in Kinshasa, Jean-Tobias Okala. (6) "There were no survivors," said Okala. (7) The spokesman said the plane, a Soviet Antonov-28 Ukrainian manufacturing and property of a company in Congo, the Congo Trasept also took a load of minerals.

Text #2 – retrieved from the newspaper *Jornal do Brasil*

(1) A crash in the town of Bukavu in the eastern Democratic Republic of Congo, killed 17 people on Thursday afternoon, reported today a spokesman of the United Nations. (2) The victims of the accident were 14 passengers and three crew members. (3) All died when the plane, hampered by bad weather, failed to reach the runway and crashed in a forest 15 km from Bukavu airport.

(4) The plane exploded and caught fire, said the spokesman of the UN in Kinshasa, Jean-Tobias Okala. (5) "There were no survivors," said Okala.

(6) The spokesman said the plane, a Soviet Antonov-28 Ukrainian manufacturing and property of a company in Congo, the Congo Trasept also took a load of minerals.

(7) According to airport sources, the crew members were of Russian nationality.

C.2 Similarity clustering

The following Tables describe the collection of sentences. The first column shows the number of the sentence; the second column refers to the number of the text from which

the sentence was retrieved; the third column details the absolute position[7] of the sentence in the Text; and the last column shows the sentence relevance score.

#	Text	Pos	Sentence	RelScore
1	2	6	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.00346
2	1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.00346
3	2	4	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.00316
4	1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.00316
5	1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.00292
6	2	7	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.00292
7	1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.00279
8	2	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.00279
9	1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	0.00274
10	2	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 Km do aeroporto de Bukavu.*	0.00274
11	2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	0.00254
12	1	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.*	0.00237
13	2	5	*'Não houve sobreviventes', disse Okala.*	0.00195
14	1	6	*'Não houve sobreviventes', disse Okala.*	0.00195

Table C.1: Sentences ordered by the relevance score **before** having been clustered by similarity.

[7] A translated version of each sentence can be found in Section C.1 from *Annex C.*

Cluster #1	2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	0.50254
	1	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.*	0.00237
Cluster #2	1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.50279
	2	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.00279
Cluster #3	1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	0.50274
	2	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 Km do aeroporto de Bukavu.*	0.00274
Cluster #4	1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.50292
	2	7	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.00292
Cluster #5	1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.50316
	2	4	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.00316
Cl.#6	1	6	*'Não houve sobreviventes', disse Okala.*	0.50195
	2	5	*'Não houve sobreviventes', disse Okala.*	0.00195
Cluster #7	1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.50346
	2	6	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.00346

Table C.2: Similarity clusters.

#	Text	Pos	Sentence	RelScore
1	1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.50346
2	1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.50316
3	1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.50292
4	1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.50279
5	1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	0.50274
6	2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	0.50254
7	1	6	*'Não houve sobreviventes', disse Okala.*	0.50195
8	2	6	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.00346
9	2	4	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.00316
10	2	7	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.00292
11	2	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.00279
12	2	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 Km do aeroporto de Bukavu.*	0.00274
13	1	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo (RDC), matou 17 pessoas na quinta-feira à tarde, informou nesta sexta-feira um porta-voz das Nações Unidas.*	0.00237
14	2	5	*'Não houve sobreviventes', disse Okala.*	0.00195

Table C.3: Sentences ordered by the relevance score **after** having been clustered by similarity.

#	Text	Pos	Sentence	RelScore
1	1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.50346
2	1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.50316
3	1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.50292
4	1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.50279
5	1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	0.50274
6	2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	0.50254
7	1	6	*'Não houve sobreviventes', disse Okala.*	0.50195

Table C.4: Representative sentences from the similarity clusters ordered by *relevance score*.

C.3 Keyword clustering

Keyword	RelScore
avião	0.00419
porta-voz	0.00419
acidente	0.00279
Bukavu	0.00279
Congo	0.00279

Table C.5: Keywords obtained for the texts in Examples 3.1 and 3.2.

In the following tables, the first column refers to the number of the text from which the sentence was retrieved; the second column details the absolute position[8] of the sentence in the Text; and the last column shows the sentence relevance score (obtained using Equation 3.2).

Text	Pos	Sentence	RelScore
2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	0.57397
1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	0.56983
1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	0.55902
1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	0.54446
1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	0.53978
1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	0.50292
1	6	*'Não houve sobreviventes', disse Okala.*	0.50197

Table C.6: Sentences ordered by *relevance score* after the score of each keyword has been updated.

[8] A translated version of each sentence can be found in Section C.1 from *Annex C.*

AVIÃO

1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	3.03978
1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	2.56983
1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	3.55902

PORTA-VOZ

2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	5.073974

BUKAVU

ACIDENTE

1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	2.04446

CONGO

NO-KEYWORD

1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	-0.49708
1	6	*'Não houve sobreviventes', disse Okala.*	-0.49804

Table C.7: Keyword clusters.

Text	Pos	Sentence	RelScore
2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	5.07397
1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	3.55902
1	3	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	3.03978
1	5	*O avião explodiu e se incendiou, acrescentou o porta-voz da ONU em Kinshasa, Jean-Tobias Okala.*	2.56983
1	2	*As vítimas do acidente foram 14 passageiros e três membros da tripulação.*	2.04446
1	4	*Segundo fontes aeroportuárias, os membros da tripulação eram de nacionalidade russa.*	-0.49707
1	6	*'Não houve sobreviventes', disse Okala.*	-0.49804

Table C.8: Sentences ordered by *relevance score* after the keyword clustering step.

Text	Pos	Sentence	RelScore
2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	5.07397
1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	3.55902

Table C.9: Sentences composing a summary with a default compression rate of 70%.

C.4 Sentence reduction

In the following tables, the first column refers to the position of the sentence in the power set; the second column shows the reduced sentence; the third column describes the phrase (or phrases) that has been removed from the original sentence; and the last column shows the sentence relevance score (obtained using Equation 3.2).

#	Sentence	Phrase reduced	RelScore
1	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	–	4.05902
2	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, também levava uma carga de minerais.*	*a Trasept Congo*	4.04527

Table C.10: Reduction power set for the sentence: *"O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais."*

#	Sentence	Phrase reduced	RelScore
1	*Todos morreram quando o avião não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	*prejudicado pelo mau tempo*	3.54645
2	*Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	–	3.53978

Table C.11: Reduction power set for the sentence: *"Todos morreram quando o avião, prejudicado pelo mau tempo, não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu."*

In the following table, the first column refers to the number of the text from which the sentence was retrieved; the second column shows the absolute position[9] of the sentence in the Text; and the last column shows the sentence relevance score (obtained using Equation 3.2).

[9] A translated version of each sentence can be found in Section C.1 from *Annex C.*

Text	Pos	Sentence	RelScore
2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	5.07397
1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	3.55902
1	3	*Todos morreram quando o avião não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	3.04645

Table C.12: Sentences ordered by *relevance score* after the sentence reduction procedure.

C.5 Paragraph creation

In the following tables, the first column refers to the number of the text from which the sentence was retrieved; the second column details the absolute position[10] of the sentence in the Text; and the last column shows the sentence relevance score (obtained using Equation 3.2). Finally, the score of the paragraph is shown, along with the keyword representing each cluster.

PORTA-VOZ (5.07397)

2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	5.07397

AVIÃO (3.30274)

1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	3.55902
1	3	*Todos morreram quando o avião não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	3.04645

Table C.13: Paragraphs composed by the sentences to be included in the final summary (the keywords are only shown for reference, as they will not be included in the summary).

[10] A translated version of each sentence can be found in Section C.1 from *Annex C.*

		PORTA-VOZ	(5.07397)
2	1	*Um acidente aéreo na localidade de Bukavu, no leste da República Democrática do Congo, matou 17 pessoas na quinta-feira à tarde, informou hoje um porta-voz das Nações Unidas.*	5.07397

		AVIÃO	(3.30274)
1	7	*O porta-voz informou que o avião, um Soviet Antonov-28 de fabricação ucraniana e propriedade de uma companhia congolesa, a Trasept Congo, também levava uma carga de minerais.*	3.55902
1	3	*Assim, todos morreram quando o avião não conseguiu chegar à pista de aterrissagem e caiu numa floresta a 15 quilômetros do aeroporto de Bukavu.*	3.04645

Table C.14: Final summary after discourse connectives have been inserted (the keywords are only shown for reference, as they will not be included in the summary).

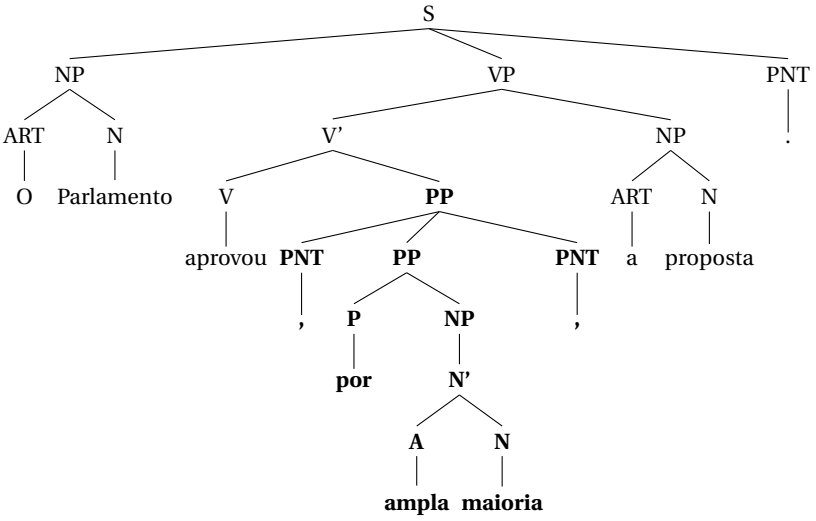
D Post-processing procedure

D.1 Sentence reduction

D.1.1 Parenthetical phrase

O Parlamento aprovou, por ampla maioria, a proposta.

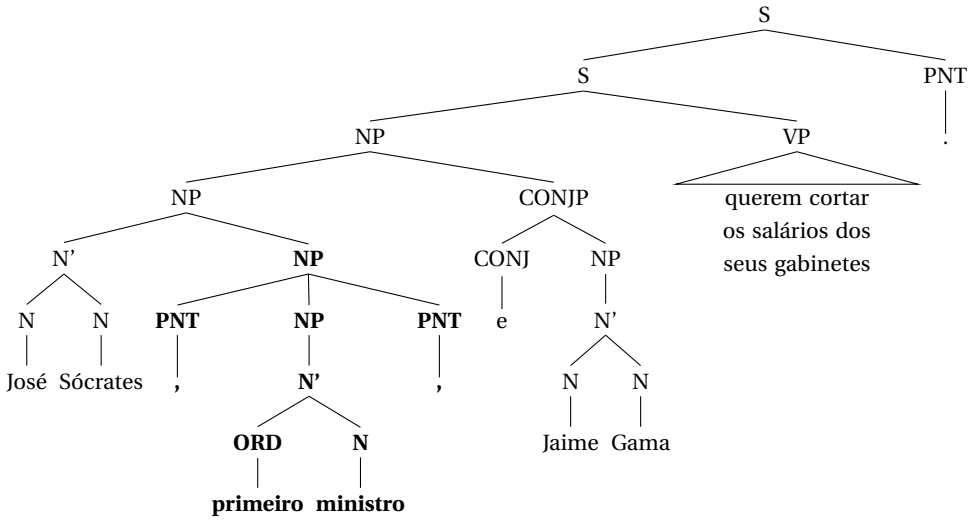
The Parliament approved by large majority the proposal.



D.1.2 Apposition

José Sócrates, primeiro-ministro, e Jaime Gama querem cortar os salários dos seus gabinetes.

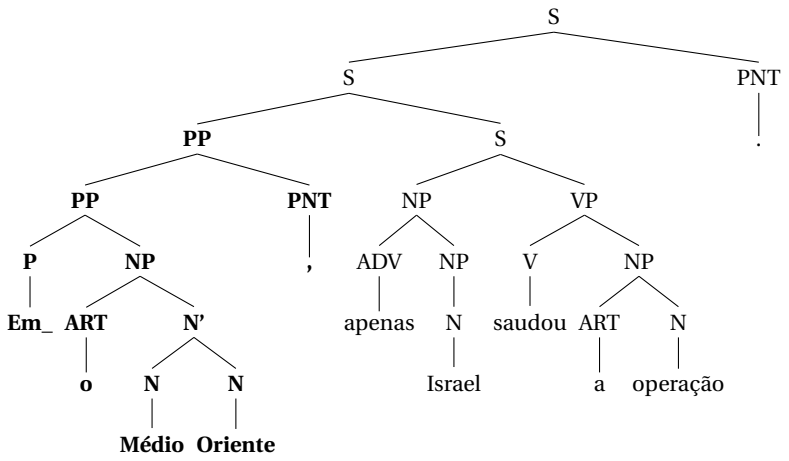
José Sócrates, the Prime Minister, and Jaime Gama want to cut the salaries of their offices.



D.1.3 Prepositional phrase

No Médio Oriente, apenas Israel saudou a operação.

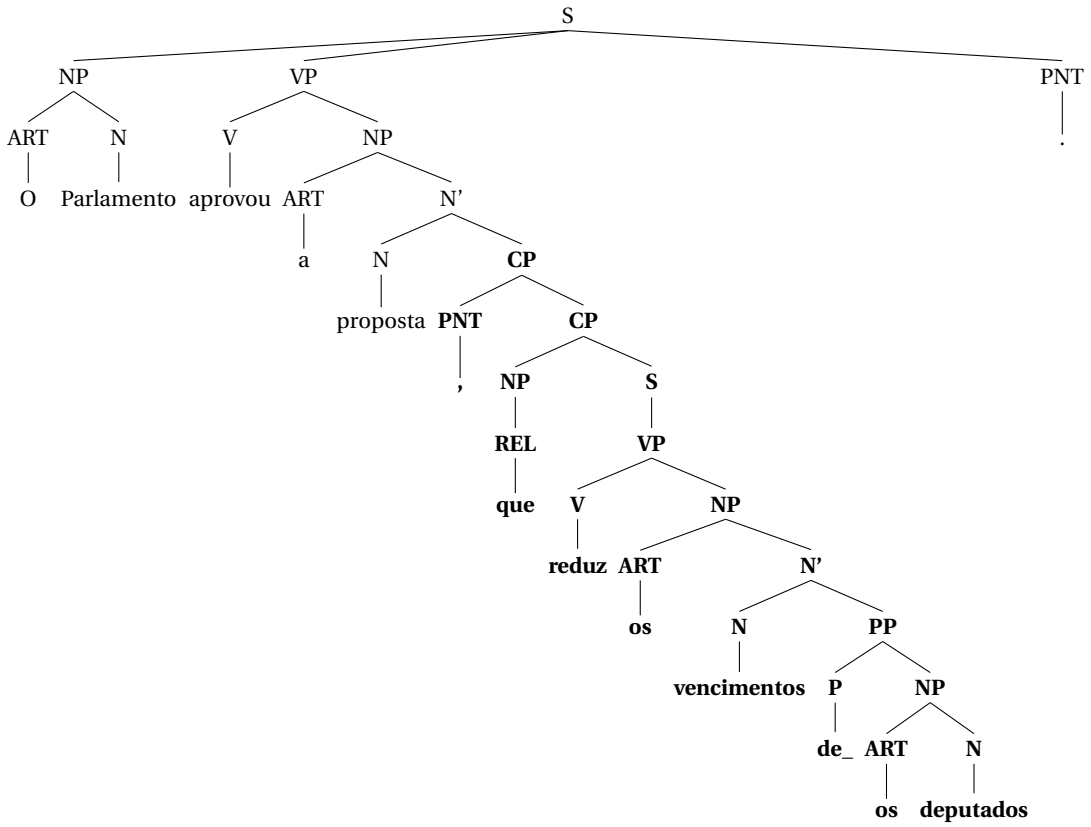
In the Middle East, only Israel welcomed the operation.



D.1.4 Relative clause

O Parlamento aprovou a proposta, que reduz os vencimentos dos deputados.

The Parliament approved the proposal, which reduces the salaries of deputies.



D.2 Discourse corpus creation

D.2.1 List of connectives

Class	Type	Subtype	Connective	Translation
COMPARISON	CONTRAST	OPPOSITION	*ainda assim* *contrariamente* *contudo* *mas* *mesmo assim* *no entanto* *pelo contrário* *porém* *por oposição* *todavia*	but, contrary, however, nevertheless, ...
	CONCESSION	EXPECTATION	*ainda que* *apesar de* *apesar de que* *embora* *mesmo que* *nem que* *por mais que* *posto que* *se bem que*	although, albeit, even, since, though, ...
		CONTRA-EXPECTATION	*como* *como se*	as, as though

Table D.15: Connectives for the class: COMPARISON.

Class	Type	Subtype	Connective	Translation
CONTINGENCY	CAUSE	REASON	*dado que* *já que* *pelo que* *pois* *pois que* *por* *por causa de* *porque* *uma vez que* *visto que*	as, because, since, whereby, ...
		RESULT	*a fim de que* *assim* *daí* *daí que* *de modo que* *desta forma* *deste modo* *em consequência* *então* *é que* *como resultado* *como tal* *consequentemente* *logo* *nesse caso* *para* *para que* *pois* *por conseguinte* *por consequência* *por esta razão* *por isso* *portanto* *posto isto*	as result, as such, because, consequently, ergo, for this reason, hence, hereupon, in order that, in this case, so, so that, this way, thence, thus, therefore, to, ...

Table D.16: Connectives for the class: CONTINGENCY (part one).

Class	Type	Subtype	Connective	Translation
CONTINGENCY	CONDITION	FACTUAL	*caso* *se*	if
		HYPOTHETICAL	*admitindo que* *a menos que* *a não ser que* *caso* *como que* *contando que* *desde que* *excepto se* *na condição de* *no caso de* *salvo se* *se* *sem que* *se porventura* *supondo que*	assuming that, if, if indeed, provided that, such that, unless, whereas, without, ...
		CONTRA-FACTUAL	*caso* *se*	if

Table D.17: Connectives for the class: CONTINGENCY (part two).

Class	Type	Subtype	Connective	Translation
TEMPORAL	ASYNCHRONOUS	PRECEDENCE	*antes de* *antes que*	before
		SUCCESSION	*depois de* *depois que*	after
	SYNCHRONOUS		*assim que* *até que* *desde que* *enquanto* *enquanto que* *logo que* *mal* *quando* *sempre que*	as soon as, until, when, whenever, while, ...

Table D.18: Connectives for the class: TEMPORAL.

Class	Type	Subtype	Connective	Translation
EXPANSION	RESTATEMENT	SPECIFICATION-EQUIVALENCE	*como seja* *de facto* *em outras palavras* *em resumo* *em suma* *isto é* *mais precisamente* *ou melhor* *ou seja* *por outras palavras* *quer dizer* *quer isto dizer*	in fact, in other words, in short, meaning, more precisely, or rather, such as, that is, this means, …
		GENERALIZATION	*afinal de contas* *concluindo* *em conclusão* *em síntese* *enfim*	after all, concluding, in conclusion, in summary, ultimately, …
	ADDITION		*adicionalmente* *além de* *assim como* *bem como* *de novo* *do mesmo modo* *e* *e ainda* *em primeiro lugar* *em seguida* *igualmente* *também*	additionally, also, and, as well as, firstly, furthermore, likewise, still, then, yet, …

Table D.19: Connectives for the class: EXPANSION (part one).

Class	Type	Subtype	Connective	Translation
EXPANSION	INSTANTIATION		*em particular* *especialmente* *especificamente* *concretamente* *nomeadamente* *particularmente* *por exemplo* *sobretudo*	concretely, for instance, mainly, namely, particularly, specifically, ...
	ALTERNATIVE	DISJUNCTIVE	*ou*	or
		CHOSEN ALTERNATIVE	*alternativamente* *em alternativa* *em vez de*	instead, alternatively, ...
	EXCEPTION		*à excepção de* *caso contrário* *com excepção de* *de outra maneira* *de outro modo* *exceptuando* *por excepção* *senão*	by exception, except, excepting, otherwise, ...

Table D.20: Connectives for the class: EXPANSION (part two).

D.2.2 Corpus statistics

D.2.3 Sentences and tokens

	Original		Filtered	
	# sentences	**# tokens**	**# sentences**	**# tokens**
Part#1	386,569	8,390,776	351,451	7,887,209
Part#2	384,015	8,329,746	348,676	7,825,378
Part#3	382,251	8,305,007	347,261	7,805,741
Part#4	380,819	8,280,712	346,004	7,779,277
Part#5	380,393	8,252,908	345,403	7,756,636
Part#6	378,510	8,206,920	344,479	7,716,720
Part#7	376,179	8,171,158	342,561	7,689,910
Part#8	376,173	8,152,983	342,012	7,676,121
Part#9	375,519	8,140,709	340,908	7,649,957
Part#10	374,667	8,114,218	341,054	7,631,018
Part#11	373,082	8,088,899	338,574	7,595,317
Part#12	371,386	8,052,252	337,709	7,566,886
Part#13	369,997	8,010,111	336,069	7,531,003
Part#14	368,850	8,009,989	334,630	7,517,673
Part#15	368,365	7,988,786	334,588	7,499,797
Part#16	366,694	7,956,521	334,262	7,482,165
Part#17	365,790	7,921,501	332,637	7,445,213
Part#18	366,272	7,923,013	332,772	7,444,183
Part#19	362,954	7,887,767	330,373	7,421,813
Part#20	216,450	4,706,199	0	0
Total	7,324,935	158,890,175	6,461,423	144,922,017
Average	366,247	7,944,509	323,071	7,246,101

Table D.21: Statistics for each Part of *CETEMPúblico*.

D.2.4 Discourse classes distribution

Type / **Subtype**	ADDITION	INSTANTIATION	RESTATEMENT		ALTERNATIVE		EXCEPTION
			SPECIFICATION-EQUIVALENCE	GENERALIZATION	DISJUNCTIVE	CHOSEN-ALTERNATIVE	
Part#1	2,735	2,524	1,887	268	34	161	201
Part#2	2,807	2,924	1,952	124	20	89	194
Part#3	2,857	2,895	1,777	251	25	281	165
Part#4	3,045	3,297	1,837	246	9	214	190
Part#5	2,944	1,832	1,948	249	18	382	177
Part#6	2,755	2,806	2,494	252	6	47	280
Part#7	2,813	2,291	1,876	188	3	107	278
Part#8	2,832	2,541	2,610	147	13	125	142
Part#9	3,118	3,082	1,673	247	79	45	154
Part#10	3,172	2,694	1,933	182	11	260	210
Part#11	2,815	2,295	1,716	268	40	42	254
Part#12	2,772	1,872	2,489	278	32	194	252
Part#13	2,700	2,579	2,099	99	35	224	116
Part#14	2,874	2,106	1,910	247	49	250	140
Part#15	2,749	2,442	1,759	147	14	370	136
Part#16	2,776	2,725	1,736	245	9	187	177
Part#17	2,839	3,806	1,663	104	40	324	129
Part#18	2,713	1,710	1,703	147	22	205	205
Part#19	2,621	2,439	2,189	255	9	93	138
Total	53,937	48,860	37,251	3,944	468	3,600	3,538

Table D.22: Number of instances found in each Part of *CETEMPúblico* for the class EXPANSION.

Type	CONTRAST	CONCESSION	
Subtype	OPPOSITION	EXPECTATION	CONTRA-EXPECTATION
Part#1	3,740	927	24
Part#2	2,975	629	88
Part#3	4,153	1,150	134
Part#4	2,728	1,417	6
Part#5	4,768	802	45
Part#6	4,224	712	103
Part#7	3,088	1,115	62
Part#8	2,817	842	102
Part#9	5,003	627	106
Part#10	2,769	477	87
Part#11	2,517	451	237
Part#12	3,218	1,743	26
Part#13	3,678	644	28
Part#14	4,996	467	97
Part#15	2,612	798	17
Part#16	2,245	804	171
Part#17	2,952	508	34
Part#18	3,386	555	30
Part#19	2,235	689	52
Total	64,104	15,357	1,449

Table D.23: Number of instances found in each Part of *CETEMPúblico* for the class COMPARISON.

Type	CAUSE		CONDITION		
Subtype	REASON	RESULT	HYPOTHETICAL	FACTUAL	CONTRA-FACTUAL
Part#1	2,259	11,927	573	5	56
Part#2	2,589	11,726	598	8	49
Part#3	2,636	11,836	671	6	43
Part#4	2,237	11,671	518	1	68
Part#5	2,979	11,669	688	5	68
Part#6	2,100	11,745	695	10	51
Part#7	2,627	11,678	616	2	45
Part#8	2,097	11,261	597	2	45
Part#9	2,677	11,510	533	2	43
Part#10	2,001	11,394	653	3	59
Part#11	1,944	11,498	528	2	54
Part#12	2,480	11,305	569	3	54
Part#13	2,558	11,411	582	3	47
Part#14	2,231	11,256	671	5	46
Part#15	2,320	11,241	643	2	56
Part#16	2,019	11,256	616	1	40
Part#17	1,855	11,034	682	1	57
Part#18	1,977	11,318	598	5	46
Part#19	2,034	11,050	700	2	57
Total	43,620	217,786	11,731	68	984

Table D.24: Number of instances found in each Part of *CETEMPúblico* for the class CONTINGENCY.

Type / **Subtype**	ASYNCHRONOUS	SYNCHRONOUS	
		PRECEDENCE	SUCCESSION
Part#1	622	2,017	195
Part#2	701	2,098	353
Part#3	588	1,983	322
Part#4	658	2,086	248
Part#5	643	2,045	388
Part#6	642	2,029	460
Part#7	614	2,014	304
Part#8	637	1,960	571
Part#9	617	2,024	265
Part#10	625	2,007	388
Part#11	622	2,033	370
Part#12	597	2,004	288
Part#13	590	1,964	209
Part#14	578	2,058	501
Part#15	601	1,971	343
Part#16	607	1,981	467
Part#17	584	1,909	180
Part#18	573	1,963	303
Part#19	624	1,979	211
Total	11,723	38,125	6,366

Table D.25: Number of instances found in each Part of *CETEMPúblico* for the class TEMPORAL.

Part	NULL
Part#1	146,826
Part#2	145,421
Part#3	142,822
Part#4	142,261
Part#5	141,184
Part#6	142,369
Part#7	142,429
Part#8	142,435
Part#9	135,531
Part#10	139,343
Part#11	142,640
Part#12	139,692
Part#13	138,416
Part#14	136,332
Part#15	139,565
Part#16	139,986
Part#17	138,310
Part#18	140,493
Part#19	139,105
Total	2,675,160

Table D.26: Number of instances found in each Part of *CETEMPúblico* for the class NULL.

Part	**Total**
Part#1	11,624
Part#2	38,194
Part#3	6,360
Part#4	43,590
Part#5	217,602
Part#6	11,730
Part#7	68
Part#8	984
Part#9	64,085
Part#10	15,342
Part#11	1,448
Part#12	53,867
Part#13	48,775
Part#14	37,237
Part#15	3,942
Part#16	467
Part#17	3,599
Part#18	3,534
Part#19	2,675,160
Total	3,237,608

Table D.27: Total number of instances found in each Part of *CETEMPúblico*.

Class	**Total**	**Percentage**
TEMPORAL-ASYNCHRONOUS-PRECEDENCE	11,624	0.36%
TEMPORAL-ASYNCHRONOUS-SUCCESSION	38,194	1.18%
TEMPORAL-SYNCHRONOUS	6,360	0.20%
CONTINGENCY-CAUSE-REASON	43,590	1.35%
CONTINGENCY-CAUSE-RESULT	217,602	6.72%
CONTINGENCY-CONDITION-HYPOTHETICAL	11,730	0.36%
CONTINGENCY-CONDITION-FACTUAL	68	0.00%
CONTINGENCY-CONDITION-CONTRA-FACTUAL	984	0.03%
COMPARISON-CONTRAST-OPPOSITION	64,085	1.98%
COMPARISON-CONCESSION-EXPECTATION	15,342	0.47%
COMPARISON-CONCESSION-CONTRA-EXPECTATION	1,448	0.04%
EXPANSION-ADDITION	53,867	1.66%
EXPANSION-INSTANTIATION	48,775	1.51%
EXPANSION-RESTATEMENT-SPECIFICATION-EQUIVALENCE	37,237	1.15%
EXPANSION-RESTATEMENT-GENERALIZATION	3,942	0.12%
EXPANSION-ALTERNATIVE-DISJUNCTIVE	467	0.01%
EXPANSION-ALTERNATIVE-CHOSEN-ALTERNATIVE	3,599	0.11%
EXPANSION-EXCEPTION	3,534	0.11%
NULL	2,675,160	82.63%
Total	3,237,608	100.00%

Table D.28: Total number of pairs found for each class in the corpus *CETEMPúblico*.

Class	**Total**	**Percentage**
TEMPORAL-ASYNCHRONOUS-PRECEDENCE	11,624	2.07%
TEMPORAL-ASYNCHRONOUS-SUCCESSION	38,194	6.79%
TEMPORAL-SYNCHRONOUS	6,360	1.13%
CONTINGENCY-CAUSE-REASON	43,590	7.75%
CONTINGENCY-CAUSE-RESULT	217,602	38.69%
CONTINGENCY-CONDITION-HYPOTHETICAL	11,730	2.09%
CONTINGENCY-CONDITION-FACTUAL	68	0.01%
CONTINGENCY-CONDITION-CONTRA-FACTUAL	984	0.17%
COMPARISON-CONTRAST-OPPOSITION	64,085	11.39%
COMPARISON-CONCESSION-EXPECTATION	15,342	2.73%
COMPARISON-CONCESSION-CONTRA-EXPECTATION	1,448	0.26%
EXPANSION-ADDITION	53,867	9.58%
EXPANSION-INSTANTIATION	48,775	8.67%
EXPANSION-RESTATEMENT-SPECIFICATION-EQUIVALENCE	37,237	6.62%
EXPANSION-RESTATEMENT-GENERALIZATION	3,942	0.70%
EXPANSION-ALTERNATIVE-DISJUNCTIVE	467	0.08%
EXPANSION-ALTERNATIVE-CHOSEN-ALTERNATIVE	3,599	0.64%
EXPANSION-EXCEPTION	3,534	0.63%
Total	562,448	100.0%

Table D.29: Percentage of discourse relation pairs found in *CETEMPúblico* without considering the NULL class.

D.2.5 Datasets statistics

Class	**Training**		**Testing**	
	Total	*Percentage*	*Total*	*Percentage*
TEMPORAL-ASYNCHRONOUS-PRECEDENCE	11,010	0.36%	614	0.35%
TEMPORAL-ASYNCHRONOUS-SUCCESSION	36,170	1.18%	2,024	1.14%
TEMPORAL-SYNCHRONOUS	6,166	0.20%	194	0.11%
CONTINGENCY-CAUSE-REASON	41,331	1.35%	2,259	1.28%
CONTINGENCY-CAUSE-RESULT	205,688	6.72%	11,914	6.73%
CONTINGENCY-CONDITION-HYPOTHETICAL	11,157	0.36%	573	0.32%
CONTINGENCY-CONDITION-FACTUAL	63	0.00%	5	0.00%
CONTINGENCY-CONDITION-CONTRA-FACTUAL	928	0.03%	56	0.03%
COMPARISON-CONTRAST-OPPOSITION	60,345	1.97%	3,740	2.11%
COMPARISON-CONCESSION-EXPECTATION	14,416	0.47%	926	0.52%
COMPARISON-CONCESSION-CONTRA-EXPECTATION	1,424	0.05%	24	0.01%
EXPANSION-ADDITION	51,135	1.67%	2,732	1.54%
EXPANSION-INSTANTIATION	46,256	1.51%	2,519	1.42%
EXPANSION-RESTATEMENT-SPECIFICATION-EQUIVALENCE	35,351	1.16%	1,886	1.07%
EXPANSION-RESTATEMENT-GENERALIZATION	3,675	0.12%	267	0.15%
EXPANSION-ALTERNATIVE-DISJUNCTIVE	433	0.01%	34	0.02%
EXPANSION-ALTERNATIVE-CHOSEN-ALTERNATIVE	3,438	0.11%	161	0.09%
EXPANSION-EXCEPTION	3,334	0.11%	200	0.11%
NULL	2,528,334	82.61%	146,826	82.97%
Total	3,060,654		176,954	

Table D.30: Total number of pairs for each class in the training dataset.

D.2.6 Disambiguation rules

D.2.7 Corpus creation

There are two types of rules: **before** and **after** rules. Depending on the connective, it must verify either a before rule, or an after rule, or both.

The following connectives:

adicionalmente, ainda assim, ainda que, além de, antes de, assim, como resultado, como tal, concretamente, consequentemente, de novo, de outro modo, de outra maneira, desta forma, deste modo, do mesmo modo, em consequência, em outras palavras, em primeiro lugar, em seguida, então, igualmente, logo, mesmo assim, particularmente, por consequência, por conseguinte, por esta razão, por isso, por outras palavras, portanto, se, sobretudo, também

must verify both the following before and after rules:

Before rule	**After rule**
– must be preceded by a comma or a semicolon.	– must be followed by an inflected verb.

The connectives:

antes que, depois de, depois que, isto é, ou seja, pelo contrário, pois, por oposição, por excepção, quer dizer

must verify a **before** rule that states that these connectives "must be preceded by a comma or a semicolon".

The connectives:

em conclusão, em vez de, mais precisamente

must verify an **after** rule that states that these connectives "must be followed by an inflected verb".

The following connectives have each one a specific rule, either before or after rule.

Connective	Before rule	After rule
assim que		– cannot be followed by a verb in the indicative.
até que		– cannot be immediately followed by a personal pronoun, a common noun or a proper name.
caso	– must be preceded by a comma or a semicolon.	– must be followed by a verb in the conjunctive.
como	– must be preceded by an inflected verb	– cannot be followed by a punctuation token or a common noun or a proper noun. – must be followed by an inflected verb.
como que		– cannot be followed by a verb in the indicative or in the infinitive.
de facto	– must be preceded by an inflected verb.	– must be immediately followed by an inflected verb.
e	– must initiate the sentence.	– must be followed by an inflected verb.
enquanto		– cannot be immediately followed by a punctuation token; – cannot be immediately preceded by *"por"* marked as a preposition; – cannot be immediately followed by a personal pronoun, a common noun or a proper name; – must be followed by an inflected verb.
enquanto que		– cannot be immediately followed by a punctuation token; – cannot be immediately followed by a common noun or a proper name; – must be followed by an inflected verb.
ou	– must initiate the sentence.	– must be followed by an inflected verb.
para	– must be preceded by a comma or a semicolon.	– must be followed by a verb in the infinitive.
por	– cannot be preceded by an inflected verb, or past participle or gerund.	– must be followed by a verb in the infinitive.
nesse caso	– must initiate the sentence.	– must be followed by an inflected verb.

Table D.31: Specific cases: rules for finding connectives in context.

D.2.8 Classification

Class	**ARG$_2$ rule**
CONTINGENCY-CONDITION-HYPOTHETICAL	– the verb must be in the subjunctive.
TEMPORAL-SYNCHRONOUS	– the verb must be in the indicative.

Table D.32: Rules applied to ARG$_2$ of a pair containing *"desde que"*.

Class	**ARG$_1$ rule**	**ARG$_2$ rule**
CONTINGENCY-CONDITION-HYPOTHETICAL	– verb in the indicative.	– verb in the subjunctive.
CONTINGENCY-CONDITION-FACTUAL	– verb in the indicative.	– verb in the indicative.
CONTINGENCY-CONDITION-CONTRA-FACTUAL	– verb in the indicative imperfect past or conditional. – verb in the indicative past perfect or conditional past.	– verb in the subjunctive imperfect past. – verb in the subjunctive past perfect.

Table D.33: Rules applied to ARG$_1$ and ARG$_2$ of a pair containing *"se"*.

Class	**ARG$_1$ rule**	**ARG$_2$ rule**
CONTINGENCY-CONDITION-HYPOTHETICAL	– verb in the indicative present or future. – verb in the indicative past perfect or conditional.	– verb in the subjunctive present. – verb in the subjunctive past perfect.
CONTINGENCY-CONDITION-FACTUAL	– verb in the indicative present.	– verb in the subjunctive future.
CONTINGENCY-CONDITION-CONTRA-FACTUAL	– verb in the indicative imperfect past or conditional. – verb in the indicative past perfect or indicative past.	– verb in the subjunctive imperfect past. – verb in the subjunctive past perfect.

Table D.34: Rules applied to ARG$_1$ and ARG$_2$ of a pair containing *"caso"*.

E Automatic evaluation

E.1 *CSTNews* statistics

This Section presents the statistics for each set of documents in the corpus *CSTNews*, identified as C*x*.

Set	Documents	Sentences	Words
C1	3	24	432
C2	3	51	998
C3	3	51	1,263
C4	3	40	847
C5	2	24	574
C6	3	37	930
C7	2	23	587
C8	3	24	600
C9	3	39	1,009
C10	3	38	964
C11	3	56	987
C12	3	34	974
C13	3	37	962
C14	3	25	739
C15	3	26	565
C16	3	47	1,034
C17	2	53	964
C18	3	72	1,301
C19	2	17	299
C20	3	45	956
C21	3	41	870
C22	3	50	964
C23	2	25	572
C24	3	24	546
C25	3	88	1,563

Set	Documents	Sentences	Words
C26	3	67	1,409
C27	3	93	1,547
C28	3	35	718
C29	3	53	1,170
C30	3	46	1,135
C31	2	10	217
C32	3	67	1,328
C33	3	77	1,637
C34	3	62	1,140
C35	3	36	879
C36	3	78	1,357
C37	2	27	475
C38	3	28	535
C39	3	36	914
C40	3	34	745
C41	3	45	958
C42	2	49	1,066
C43	3	51	1,267
C44	2	35	736
C45	3	67	1,226
C46	3	39	753
C47	3	48	1,373
C48	2	52	799
C49	3	53	1,001
C50	3	68	1,549

	Documents	Sentences	Words
Total	140	2,247	47,434
Average	2.8	44.94	948.68

Table E.35: Number of documents, sentences and words in the corpus *CSTNews*.

E.2 Post-processing statistics

These tables contain the number of words (and sentences when indicated) that are involved in the post-processing procedure. Considering sentence reduction, the number of words reported was removed. Otherwise, regarding connective insertion and paragraph creation, the number of words stated was inserted.

Set	**Before Post-processing**		**After Post-processing**			
			COMPRESSION TASKS		FLUENCY TASKS	
	Sentences	*Words*	*Sentences*	*Words*	*Sentences*	*Words*
C1	3	70	4	83	4	85
C2	6	174	4	166	4	170
C3	5	201	5	188	5	192
C4	3	126	8	128	8	132
C5	4	113	5	113	5	117
C6	6	196	6	185	6	190
C7	5	118	4	115	4	118
C8	3	112	4	111	4	115
C9	3	162	6	169	6	173
C10	5	163	7	146	7	149
C11	8	156	8	154	8	159
C12	5	165	2	165	2	167
C13	6	165	4	156	4	160
C14	4	139	6	132	6	134
C15	4	88	6	85	6	87
C16	5	157	7	172	7	175
C17	5	157	3	157	3	162
C18	8	200	10	197	10	201
C19	2	49	7	49	7	50
C20	5	148	10	157	10	165
C21	5	148	3	142	3	146
C22	5	161	5	166	5	170
C23	3	93	6	92	6	93
C24	3	84	6	105	6	109
C25	8	259	2	246	2	255

Table E.36: Number of sentences and words involved in the post-processing procedure (part1).

Set	Before Post-processing		After Post-processing			
			COMPRESSION TASKS		FLUENCY TASKS	
	Sentences	*Words*	*Sentences*	*Words*	*Sentences*	*Words*
C26	7	234	8	212	8	214
C27	9	237	8	259	8	269
C28	3	124	7	118	7	121
C29	5	197	5	185	5	187
C30	5	197	11	171	11	174
C31	2	64	4	53	4	55
C32	7	204	4	218	4	224
C33	7	246	6	262	6	265
C34	6	179	3	191	3	195
C35	4	133	7	132	7	137
C36	9	210	5	226	5	233
C37	3	71	5	86	5	90
C38	3	81	4	108	4	110
C39	3	146	9	150	9	152
C40	3	126	6	116	6	119
C41	5	158	7	163	7	168
C42	4	160	9	169	9	170
C43	5	214	6	194	6	200
C44	3	112	4	143	4	147
C45	8	194	7	205	7	212
C46	5	132	4	133	4	139
C47	5	206	6	238	6	241
C48	7	123	5	128	5	133
C49	6	160	3	160	3	165
C50	5	236	4	243	4	248
Total	248	7748	285	7842	285	8042
Average	4.96	154.96	5.7	156.84	5.7	160.84

Table E.37: Number of sentences and words involved in the post-processing procedure (part2).

F Human evaluation

F.1 Surveys

F.1.1 Sentence reduction

The full survey for this module can be found in `https://docs.google.com/forms/d/1-hC45e3UeEPfZNatYq4GR7iee7AprfsfLbXlsLJ4jBM/viewform` (in Portuguese).

Task#1

Text#1

A pista principal do Aeroporto Internacional de São Paulo em Guarulhos, será totalmente reformada a partir de março de 2008, segundo informações do Ministério da Defesa. A reforma emergencial, foi descartada. Será possível realizar a obra em três etapas. Na primeira etapa, será reformado um terço da pista, em uma das cabeceiras, ficando o restante disponível para pousos e decolagens. Na segunda etapa, a parte concluída será reaberta e a obra passará a ser feita na outra cabeceira. Será reformado o trecho central, quando a pista será fechada e os vôos que a utilizariam serão transferidos para o Aeroporto de Viracopos, em Campinas. O cronograma final da obra depende de estudos que estão sendo realizados pela Infraero. A transferência dos vôos de Guarulhos para Viracopos não poderá ser feita neste momento porque o aeroporto de Campinas necessitará de ampliação.

Text#2

A pista principal do Aeroporto Internacional de São Paulo (Cumbica), em Guarulhos, será totalmente reformada a partir de março de 2008, segundo informações do Ministério da Defesa. Com isso, a reforma emergencial, que começaria em breve, foi descartada. De acordo com a Infraero, será possível realizar a obra em três etapas. Na primeira etapa, será reformado um terço da pista, em uma das cabeceiras, ficando o restante disponível para pousos e decolagens. Na segunda etapa, a parte concluída será reaberta e a obra passará a ser feita na outra cabeceira. Por fim, será reformado o trecho central, quando a pista será fechada e os vôos que a utilizariam serão transferidos para o Aeroporto de Viracopos, em Campinas. O cronograma final da obra depende de estudos que estão sendo realizados pela Infraero. A transferência dos vôos de Guarulhos para Viracopos não poderá ser feita neste momento porque o aeroporto de Campinas necessitará de ampliação.

Questions

1. Which text is easier to read and comprehend (Text#1 or Text#2)?
2. Which text is organized in a more effective way (Text#1 or Text#2)?

3. How do you classify the textual quality of Text#1? (0-5)
4. How do you classify the textual quality of Text#2? (0-5)
5. Commentary – Please insert here your commentary if you find it appropriate.

F.1.2 Paragraph creation

The full survey for this module can be found in `https://docs.google.com/forms/d/1YuO9nn2MbksDs1rm34WHhQM7ZAosavcMYaWUA2BXcBQ/viewform` (in Portuguese).

Task#1

Text#1

17 pessoas morreram após a queda de um avião na República Democrática do Congo. O avião saiu de Lugushwa a Bukavu e caiu sobre uma floresta após se chocar com uma montanha, prejudicado pelo mau tempo. O avião também levava cargas e minerais.
14 dessas vítimas eram passageiros e três membros da tripulação, todos de nacionalidade russa. Nenhuma vítima sobreviveu.

Text#2

17 pessoas morreram após a queda de um avião na República Democrática do Congo. 14 dessas vítimas eram passageiros e três membros da tripulação, todos de nacionalidade russa. Nenhuma vítima sobreviveu. O avião saiu de Lugushwa a Bukavu e caiu sobre uma floresta após se chocar com uma montanha, prejudicado pelo mau tempo. O avião também levava cargas e minerais.

Questions

1. Which text is easier to read and comprehend (Text#1 or Text#2)?
2. Which text is organized in a more effective way (Text#1 or Text#2)?
3. How do you classify the textual quality of Text#1? (0-5)
4. How do you classify the textual quality of Text#2? (0-5)
5. Commentary – Please insert here your commentary if you find it appropriate.

F.1.3 Connective insertion

The full survey for this module can be found in `https://docs.google.com/forms/d/1VtphBlf246frV9Vr8jscA6KwrUuhlUnMVmlLhlZNX7Y/viewform` (in Portuguese).

Task#1

Text#1

O presidente Luiz Inácio Lula da Silva afirmou nesta segunda-feira, durante o programa de rádio 'Café com o Presidente', que vai anunciar obras de infra-estrutura e saneamento que transformarão o Brasil em um 'verdadeiro canteiro de obras'. O presidente diz que algumas das obras já estão em andamento, outras vão começar logo e para algumas outras ainda falta licenciamento. Porque, segundo o presidente, a prioridade é a realização de obras nas regiões metropolitanas de grandes centros urbanos. O presidente também afirmou que o critério para os municípios e Estados contemplados com obras é técnico.
Lula anunciou na sexta-feira R$6 bilhões em investimentos do Programa de Aceleração do Crescimento (PAC) para urbanização de favelas e saneamento básico.

Text#2

O presidente Luiz Inácio Lula da Silva afirmou nesta segunda-feira, durante o programa de rádio "Café com o Presidente", que vai anunciar obras de infra-estrutura e saneamento que transformarão o Brasil em um "verdadeiro canteiro de obras". Lula anunciou na sexta-feira R$6 bilhões em investimentos do Programa de Aceleração do Crescimento (PAC) para urbanização de favelas e saneamento básico. O presidente diz que algumas das obras já estão em andamento, outras vão começar logo e para algumas outras ainda falta licenciamento. Segundo o presidente, a prioridade é a realização de obras nas regiões metropolitanas de grandes centros urbanos. O presidente também afirmou que o critério para os municípios e Estados contemplados com obras é técnico.

Questions

1. Which text is easier to read and comprehend (Text#1 or Text#2)?
2. Which text is organized in a more effective way (Text#1 or Text#2)?
3. How do you classify the textual quality of Text#1? (0-5)
4. How do you classify the textual quality of Text#2? (0-5)
5. Commentary – Please insert here your commentary if you find it appropriate.

F.1.4 SIMBA

The full survey for this module can be found in `https://docs.google.com/forms/d/1vUDyv09XHs374Y_JpJIccl0EozlZzBwPH-6PEJ9jVK8/viewform` (in Portuguese).

Task#1

Original texts

Text#1

Equipes de resgate procuram sobreviventes em destroços de casa na cidade de Kashiwazaki

TÓQUIO - Um terremoto de 6,8 graus na escala Richter atingiu a costa noroeste do Japão nesta segunda-feira, 16, matando pelo menos sete pessoas na cidade de Kashiwazaki e deixando outros 700 feridos.

Equipes de resgate continuam procurando por pessoas em meio aos escombros. Chamas e rolos de fumaça preta foram vistos na usina nuclear de Kashiwazaki, que foi automaticamente fechada durante o terremoto.

Segundo a TV NHK, o fogo atingiu um transformador de eletricidade e não houve vazamento de radiação. A força do abalo danificou estradas e pontes na costa.

Cerca de 2 mil moradores de Kashiwazaki, a cidade mais afetada pelo tremor, tiveram de deixar suas casas. Cerca de 300 edifícios da cidade ficaram destruídos.

Rachaduras de um metro de largura foram vistas no solo ao longo do litoral. A Agência de Meteorologia divulgou alertas de tsunami para o estado de Niigata e a região costeira das redondezas, mas por fim o aviso foi cancelado.

O tremor atingiu a região às 10h13 (horário local, 22h13 de domingo, em Brasília) e seu epicentro foi localizado a 260 km da costa de Niigata, ao nordeste da capital, Tóquio, onde também foi sentido. Uma série de pequenos abalos secundários atingiu a área, dentre eles um de 4,2 graus. Segundo Koichi Uhira, da Agência de Meteorologia, esses tremores podem continuar por uma semana.

Vários serviços de trem bala ligando Tóquio ao norte e ao nordeste do país foram suspensos. Informações oficiais indicam que o fornecimento de água e gás para as 35 mil residências de Kashiwazaki foi cortado depois que foram constatados vazamentos de gás. Cerca de 27 mil famílias estavam sem energia elétrica na manhã desta segunda-feira em Niigata.

Text#2

TÓQUIO - Um terremoto de 6,8 graus na escala Richter, com epicentro a 17 quilômetros de profundidade, atingiu a costa noroeste do Japão às 10h13m desta segunda-feira (22h13m de domingo em Brasília). O tremor derrubou vários prédios. Quatro pessoas morreram e cerca de 400 ficaram feridas.

O terremoto, que pôde ser sentido em Tóquio, foi seguido por outro tremor de menor magnitude, de 4,2 graus na escala Ritcher, às 10h34m (22h34m de domingo em Brasília).

A agência meteorológica do Japão chegou a emitir um alerta de tsunami para a Ilha Sado, na costa da província de Niigata, mas suspendeu o aviso uma hora depois.

Duas mulheres de cerca de 80 anos morreram quando suas casas desmoronaram após o tremor. Os detalhes das outras duas mortes, informadas pela emissora pública NHK, não estão disponíveis.

Um pequeno incêndio aconteceu em um transformador elétrico da usina nuclear de Kashiwazaki Kariwa, a maior do mundo, localizada perto do epicentro, mas o fogo já foi controlado. Os reatores nucleares foram desligados e não houve liberação de radiação. Niigata foi atingida em outubro de 2004 por um terremoto de mesma intensidade, que provocou a morte de 65 pessoas e deixou mais de 3 mil feridos.

Text#3

KASHIWAZAKI - Um forte terremoto matou ao menos cinco pessoas no noroeste do Japão nesta segunda-feira. Mais de 600 pessoas ficaram feridas, casas foram derrubadas e houve um pequeno incêndio na maior usina nuclear do mundo.

Os prédios chegaram a tremer em Tóquio, e os reatores de usinas nucleares em Niigata desligaram-se automaticamente para checagens, embora não haja relatos de vazamento de radiação.

Duas mulheres na faixa dos 80 anos morreram quando suas casas ruíram durante o tremor de magnitude 6,8, na área de Niigata, cerca de 250 km noroeste de Tóquio, informou a imprensa japonesa.

Um porta-voz policial confirmou cinco mortes. A imprensa noticiou que uma outra idosa e um casal estariam entre os mortos.
- Prateleiras altas caíram e as coisas voaram por toda parte - contou Harumi Mikami, 55, uma professora que estava em sua escola na Cidade de Kashiwazaki, perto do epicentro do terremoto.

- Metade da minha casa está destruída - disse Mikami.

- Minha maior preocupação é onde vou viver agora.

Cerca de 1.700 pessoas fugiram de suas casas para quase 100 centros de resgate, segundo a rede NHK e a Prefeitura de Niigata.

Um incêndio em um transformador elétrico na usina nuclear de Kashiwazaki Kariwa foi rapidamente extinto, mas ainda não está claro quando a companhia elétrica de Tóquio vai religar três unidades no complexo, disse Yoshinobu Kamijima, porta-voz da empresa.

O Japão é um dos países do mundo mais suscetíveis a terremotos, com um tremor ocorrendo a ao menos cada cinco minutos.

Os mercados financeiros do país estão fechados nesta segunda-feira devido a um feriado.

Summaries

Summary#1

Duas mulheres na faixa dos 80 anos morreram quando suas casas ruíram durante o tremor de magnitude 6,8, na área de Niigata, cerca de 250 km noroeste de Tóquio, informou a imprensa japonesa. Cerca de 2 mil moradores de Kashiwazaki, a cidade mais afetada pelo tremor, tiveram de deixar suas casas. Um terremoto de 6,8 graus na escala Richter atingiu a costa noroeste do Japão nesta segunda-feira, 16, matando pelo menos sete pessoas na cidade de Kashiwazaki e deixando outros 700 feridos. Cerca de 1.700 pessoas fugiram de suas casas para quase 100 centros de resgate, segundo a rede NHK e a Prefeitura de Niigata. Niigata foi atingida em outubro de 2004 por um terremoto de mesma intensidade, que provocou a morte de 65 pessoas e deixou mais de 3 mil feridos.

Summary#2

Um terremoto de 6,8 graus na escala Richter atingiu a costa noroeste do Japão nesta segunda-feira, matando pelo menos sete pessoas na cidade de Kashiwazaki e deixando outros 700 feridos.

Duas mulheres na faixa dos 80 anos morreram quando suas casas ruíram durante o tremor de magnitude na área de Niigata, cerca de 250 km noroeste de Tóquio, informou a imprensa japonesa. Niigata foi atingida em outubro de 2004 por um terremoto de mesma intensidade. De outro modo, cerca de 1.700 pessoas fugiram de suas casas para quase 100 centros de resgate, segundo a rede NHK e a Prefeitura de Niigata.

O terremoto, que pôde ser sentido em Tóquio, foi seguido por outro tremor de menor magnitude, de 4,2 graus na escala Ritcher. Até que cerca de 2 mil moradores de Kashiwazaki, tiveram de deixar suas casas.

Questions about the summaries

1. Which text is the best summary for the input texts (Summary#1 or Summary#2)?
2. Considering the input texts, how much relevant information each summary contains? (0-5)
3. How much repeated information each summary contains? (0-5)

Questions about the texts

1. Which text is easier to read and comprehend (Text#1 or Text#2)?
2. Which text is organized in a more effective way (Text#1 or Text#2)?
3. How do you classify the textual quality of Text#1? (0-5)
4. How do you classify the textual quality of Text#2? (0-5)
5. Commentary – Please insert here your commentary if you find it appropriate.

F.2 E-mail

Hi!

In the context of my PhD work in Computer Science, my research is focused in Automatic Summarization. In order to proceed, I am carrying out an evaluation process involving human users. I need your collaboration in this process.

This evaluation aims to assess the quality of the texts presented. This assessment is performed by answering a survey, which is here: <link>

Please answer this survey. For the answer to be valid, the survey must be answered without interruptions. In addition, while answering to the questions, please give your first opinion. There are no correct or incorrect answers. Your opinion is what matters.

This task will take at most 20 minutes. Please reply to this message as soon as you have completed the survey. Thank you the time spent in this task, which will be very useful for my research.

If you can send this link to someone you know, I am very grateful: every help is welcome!

Thank you very much!

REFERENCES

Aleixo, P. and Pardo, T. A. S. (2008). CSTNews: Um córpus de textos jornalísticos anotados segundo a teoria discursiva multidocumento CST (cross-document structure theory). Technical report, Universidade de São Paulo.

Aluísio, S. M., Specia, L., Pardo, T. A., Maziero, E. G., and Fortes, R. P. M. (2008). Towards brazilian portuguese automatic text simplification systems. In *Proceedings of the 8th Association for Computing Machinery Conference Symposium on Document Engineering (DocEng 2008)*, pages 240–248, New York, NY, USA. Association for Computing Machinery.

Antiqueira, L. (2007). Desenvolvimento de técnicas baseadas em redes complexas para sumarização extrativa de textos. Master's thesis, Instituto de Ciências Matmáticas e de Computação - Universidade de São Paulo.

Antiqueira, L., Oliveira-Jr., O. N., da Fontoura Costa, L., and das Graças Volpe Nunes, M. (2009). A complex network approach to text summarization. *Information Sciences: an International Journal*, 179(5):584–599.

Arora, R. and Ravindran, B. (2008). Latent dirichlet allocation based multi-document summarization. In *Proceedings of the 2nd Workshop on Analytics for Noisy Unstructured Text Data (AND 2008)*, volume 303, pages 91–97, New York, NY, USA. Association for Computing Machinery.

Balage Filho, P. P. and Pardo, T. A. S. (2007). Summarizing scientific texts: Experiments with extractive summarizers. pages 520–524.

Barzilay, R. and Elhadad, M. (1997). Using lexical chains for text summarization. In *Proceedings of the Association of Computational Linguistics – Workshop on Intelligent Scal-*

able Text Summarization (ACL 1997), pages 10–17. Association for Computational Linguistics.

Barzilay, R., Elhadad, N., and McKeown, K. R. (2002). Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17(1):35–55.

Barzilay, R., McKeown, K., and Elhadad, M. (1999). Information fusion in the context of multi-document summarization. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics (ACL 1999)*, pages 550–557, Stroudsburg, PA, USA. Association for Computational Linguistics.

Berg-Kirkpatrick, T., Gillick, D., and Klein, D. (2011). Jointly learning to extract and compress. In *Proceedings of the 11th Conference on Human Language Technology Research (HTL 2011)*, pages 481–490, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bikel, D. M., Schwartz, R., and Weischedel, R. M. (1999). An Algorithm that Learns What's in a Name. *Machine Learning – Special Issue on Natural Language Learning*, 34(1–3):211–231.

Biran, O. and Rambow, O. (2011). Identifying justifications in written dialogs by classifying text as argumentative. *International Journal of Semantic Computing*, 5(4):363–381.

Blair-Goldensohn, S., McKeown, K., and Rambow, O. (2007). Building and refining rhetorical-semantic relation models. In *Proceedings of the 7th Conference on Human Language Technology Research (HLT 2007)*, pages 428–435, Rochester, New York. Association for Computational Linguistics.

Blei, D. M., Griffiths, T. L., Jordan, M. I., and Tenenbaum, J. B. (2004). Hierarchical topic models and the nested chinese restaurant process. In *Proceedings of the Conference Advances in Neural Information Processing Systems (NIPS 2004)*. MIT Press.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bollegala, D., Okazaki, N., and Ishizuka, M. (2006). A bottom-up approach to sentence ordering for multi-document summarization. In *Proceedings of the 21st Conference on*

Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (ACL 2006), pages 385–392, Stroudsburg, PA, USA. Association for Computational Linguistics.

Branco, A., Costa, F., Ferreira, E., Martins, P., Nunes, F., Silva, J., and Silveira, S. (2009). LX–center: a center of online linguistic services. In *Proceedings of the 4th International Joint Conference of Natural Language Processing (IJCNLP 2009)*, Singapore.

Branco, A., Costa, F., Martins, P., Nunes, F., Silva, J., and Silveira, S. (2008). Lxservice: Web services of language technology for portuguese. In *Proceedings of the 6th Language Resources and Evaluation Conference (LREC 2008)*, Paris.

Branco, A., Costa, F., Silva, J., Silveira, S., Castro, S., Avelãs, M., Pinto, C., and Graça, J. (2010). Developing a deep linguistic databank supporting a collection of treebanks: the CINTIL DeepGramBank. In *Proceedings of the 7th Language Resources and Evaluation Conference (LREC 2010)*, pages 1810–1815.

Branco, A. and Silva, J. (2006). A suite of shallow processing tools for portuguese: Lx-suite. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*.

Brandow, R., Mitze, K., and Rau, L. F. (1995). Automatic condensation of electronic publications by sentence selection. *Information Processing and Management: an International Journal - Special issue: Summarizing Text*, 31(5):675–685.

Brants, T. (2000). TnT — a statistical part-of-speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference and the 1st North American Chapter of the Association for Computational Linguistics (ANLP-NAACL 2000)*, pages 224–231.

Candido Jr., A., de Oliveira, M., and Aluísio, S. M. (2009). Simplifica: a simplified texts web authoring system. In *Proceedings of the XV Brazilian Symposium on Multimedia and the Web (WebMedia 2009)*, pages 46:1–46:4, New York, NY, USA. Association for Computing Machinery.

Carbonell, J. G. and Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Conference*

on Research and Development in Information Retrieval (SIGIR 1998), pages 335–336, New York, NY, USA. Association for Computing Machinery.

Carlson, L., Marcu, D., and Okurowski, M. E. (2001). Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue - Volume 16 (SIGDIAL 2001)*, pages 1–10, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chandrasekar, R., Doran, C., and Srinivas, B. (1996). Motivations and methods for text simplification. In *Proceedings of the 16th Conference on Computational Linguistics (COLING 1996)*, pages 1041–1044, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chandrasekar, R. and Srinivas, B. (1997). Automatic induction of rules for text simplification. *Journal of Knowledge-Based Systems*, 10:183–190.

Cohn, T. and Lapata, M. (2007). Large margin synchronous generation and its application to sentence compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 73–82. Association for Computational Linguistics.

Cohn, T. and Lapata, M. (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Conroy, J. M. and O'Leary, D. P. (2001). Text summarization via hidden markov models. In *Research and Development in Information Retrieval*, pages 406–407.

Costa, F. (2013). *Processing Temporal Information in Unstructured Documents*. PhD thesis, Universidade de Lisboa, Lisbon.

Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

Dorr, B., Zajic, D., and Schwartz, R. (2003). Hedge trimmer: a parse-and-trim approach to headline generation. In *Proceedings of the Conference on Human Language Technology Research and the North American Chapter of the Association for Computational Linguistics - Text Summarization Workshop Volume 5 (HLT-NAACL 2003)*, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. *Journal of the Association for Computational Linguistics*, 19(1):61–74.

Edmundson, H. P. (1969). New methods in automatic extracting. *Journal of the Association for Computational Linguistics*, 16(2):264–285.

Ercan, G. and Cicekli, I. (2008). Lexical cohesion based topic modeling for summarization. In *Proceedings of the 9th Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2008)*, pages 582–592, Berlin, Heidelberg. Springer.

Fang, Y. and Teufel, S. (2014). A summariser based on human memory limitations and lexical competition. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 732–741. Association for Computational Linguistics.

Farzindar, A., Rozon, F., and Lapalme, G. (2005). CATS: A topic-oriented multi-document summarization system. In *Proceedings of the Document Understanding Conference (DUC 2005)*, Vancouver. NIST.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. MIT Press.

Feng, L. (2008). Text simplification: A survey. Technical report, The City University of New York.

Feng, V. W. and Hirst, G. (2012). Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1 (ACL 2012)*, pages 60–68, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ferreira, E., Balsa, J., and Branco, A. (2007). Combining rule-based and statistical models for named entity recognition of Portuguese. In *Proceedings of the Workshop em Tecnologia da Informação e de Linguagem Natural (TIL 2007)*, pages 1615–1624.

Filippova, K. (2010). Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 322–330, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fuentes, M., Alfonseca, E., and Rodríguez, H. (2007). Support vector machines for query-focused summarization trained and evaluated on pyramid data. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics Conference – Interactive Poster and Demonstration Sessions (ACL 2007)*, pages 57–60, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gasperin, C., Maziero, E., Specia, L., Pardo, T., and Aluísio, S. M. (2009a). Natural language processing for social inclusion: a text simplification architecture for different literacy levels. In *Proceedings of the SEMISH-XXXVI Seminário Integrado de Software e Hardware*, pages 387–401.

Gasperin, C., Specia, L., Pereira, T., and Aluísio, S. (2009b). Learning when to simplify sentences for natural text simplification. In *Proceedings of the Encontro Nacional de Inteligência Artificial (ENIA-2009)*, pages 809–818.

Giannakopoulos, G., Karkaletsis, V., Vouros, G. A., and Stamatopoulos, P. (2008). Summarization system evaluation revisited: N-gram graphs. *Journal of the ACM Transactions on Speech and Language Processing*, 5(3):5:1–5:39.

Gonçalves, P. N. (2008). Revisão de coesão referencial em sumários extrativos. Master's thesis, Universidade do Vale do Rio dos Sinos.

Halliday, M. A. K. and Hasan, R. (1976). *Cohesion in English (English Language)*. Longman Pub Group.

Hearst, M. A. (1997). Texttiling: segmenting text into multi-paragraph subtopic passages. *Journal of the Association for Computational Linguistics*, 23(1):33–64.

Hobbs, J. R. (1985). On the Coherence and Structure of Discourse. Technical report.

Hong, K. and Nenkova, A. (2014). Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 712–721. Association for Computational Linguistics.

Hovy, E. (2004). *The Oxford Handbook of Computational Linguistics*, chapter 32, Automated Text Summarization. In Mitkov (2004).

Hovy, E. and Lin, C.-Y. (1999). *Automated Text Summarization in SUMMARIST*. MIT Press.

Jaccard, P. (1908). Nouvelles recherches sur la distribution florale. *Bulletin de la Sociète Vaudense des Sciences Naturelles*, 44:223–270.

Jing, H. (2000). Sentence reduction for automatic text summarization. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP 2000)*, pages 310–315, Morristown, NJ, USA. Association for Computational Linguistics.

Jing, H. and McKeown, K. R. (2000). Cut and paste based text summarization. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference (NAACL 2000)*, pages 178–185, Stroudsburg, PA, USA. Association for Computational Linguistics.

John, G. H. and Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the 11th conference on Uncertainty in Artificial Intelligence (UAI 1995)*, pages 338–345, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jones, K. S. (2007). Automatic summarising: The state of the art. *Information Processing and Management: an International Journal*, 43(6):1449–1481.

Jorge, M. L. C. and Pardo, T. A. S. (2012). Multi-document summarization: Content selection based on CST model (cross-document structure theory). In *Proceedings of the Encontro para o Processamento Computacional da Língua Portuguesa Escrita e Falada – PhD and MSc/MA Dissertation Contest (PROPOR 2012)*, pages 1–8, Coimbra, Portugal. Springer-Verlag.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory (Studies in Linguistics and Philosophy)*. Springer.

Kaspersson, T., Smith, C., Danielsson, H., and Jönsson, A. (2012). This also affects the context - errors in extraction based summaries. In *Proceedings of the 8th Language Resources and Evaluation Conference (LREC 2012)*, pages 173–178, Istanbul, Turkey. European Language Resources Association (ELRA).

Kintsch, W. and van Dijk, T. A. (1978). Toward a model of text comprehension and production. *Psychological Review*, 85:363–394.

Klebanov, B. B., Knight, K., and Marcu, D. (2004). Text simplification for information-seeking applications. In Meersman, R. and Tari, Z., editors, *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, volume 3290 of *Lecture Notes in Computer Science*, pages 735–747. Springer Berlin Heidelberg.

Klein, D. and Manning, C. D. (2003). Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press.

Kolluru, B. and Gotoh, Y. (2005). On the subjectivity of human authored summaries. In *Proceedings of the Association for Computational Linguistics Conference – Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization (ACL 2005)*, pages 9–16, Ann Arbor, Michigan. Association for Computational Linguistics.

Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer. In *Proceedings of the 18th Conference on Research and Development in Information Retrieval (SIGIR 1995)*, pages 68–73, New York, NY, USA. Association for Computing Machinery.

Lapata, M. (2003). Probabilistic text structuring: experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1 (ACL 2003)*, pages 545–552, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lapata, M. and Lascarides, A. (2004). Inferring sentence-internal temporal relations. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2004)*, pages 153–160, Boston.

Lascarides, A. and Asher, N. (2007). Segmented discourse representation theory: Dynamic semantics with discourse structure. In Bunt, H. and Muskens, R., editors, *Computing Meaning: Volume 3*, pages 87–124. Kluwer Academic Publishers.

Lee, A., Prasad, R., Joshi, A., and Dinesh, N. (2006). Complexity of dependencies in discourse: Are dependencies in discourse more complex than in syntax? In *Proceedings of the 5th International Workshop on Treebanks and Linguistic Theories (TLT 2006)*, page 12, Prague, Czech Republic.

Leite, D. S. and Rino, L. H. M. (2008). Combining multiple features for automatic text summarization through machine learning. In *Proceedings of the Encontro para o Processamento Computacional da Língua Portuguesa Escrita e Falada (PROPOR 2008)*, pages 122–132, Berlin, Heidelberg. Springer Verlag.

Leite, D. S., Rino, L. H. M., Pardo, T. A. S., and das Graças V. Nunes, M. (2007). Extractive automatic summarization: Does more linguistic knowledge make a difference? In *Proceedings of the 2nd Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing*, pages 17–24, Rochester, NY, USA. Association for Computational Linguistics.

Levy, R. and Andrew, G. (2006). Tregex and Tsurgeon: Tools for querying and manipulating tree data structures. In *Proceedings of the 5th Language Resources and Evaluation Conference (LREC 2006)*, page 2231–2234.

Lin, C.-Y. (2003). Improving summarization performance by sentence compression: a pilot study. In *Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages - Volume 11 (AsianIR 2003)*, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004)*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Lin, C.-Y. and Hovy, E. (1997). Identifying topics by position. In *Proceedings of the 5th Applied Natural Language Processing Conference (ANLP 1997)*, pages 283–290, Morristown, NJ, USA. Association for Computational Linguistics.

Lin, C.-Y. and Hovy, E. (2002a). Automated multi-document summarization in neats. In *Proceedings of the 2nd Conference on Human Language Technology Research (HLT 2002)*, pages 59–62, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lin, C.-Y. and Hovy, E. (2002b). From single to multi-document summarization: A prototype system and its evaluation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 457–464, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lin, C.-Y. and Hovy, E. H. (2000). The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th Conference on Computational Linguistics – Volume 1 (COLING 2000)*, pages 495–501, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lin, D. (1998). Dependency-based Evaluation of MINIPAR. In *Proceedings of the Language Resources and Evaluation Conference – Workshop on the Evaluation of Parsing Systems (LREC 1998)*.

Lin, Z., Kan, M. Y., and Ng, H. T. (2009). Recognizing implicit discourse relations in the Penn Discourse Treebank. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EMNLP 2009)*, pages 343–351, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lloret, E. (2011). *Text Summarisation based on Human Language Technologies and its Applications*. PhD thesis, Universidad de Alicante.

Louis, A., Joshi, A., and Nenkova, A. (2010). Discourse indicators for content selection in summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2010)*, pages 147–156, Stroudsburg, PA, USA. Association for Computational Linguistics.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

Mani, I. (2001a). *Automatic Summarization*. John Benjamins Publishing Company, Amsterdam.

Mani, I. (2001b). Summarization evaluation: An overview. In *Proceedings of the 2nd NII Testbeds and Community for Information Access Research Workshop on Research in Chinese & Japanese Text Retrieval and Text Summarization (NTCIR 2001)*.

Mani, I. and Bloedorn, E. (1999). Summarizing similarities and differences among related documents. *Journal of Information Retrieval*, 1(1):35–67.

Mani, I., Klein, G., House, D., Hirschman, L., Firmin, T., and Sundheim, B. (2002). SUMMAC: a text summarization evaluation. *Journal of Natural Language Engineering*, 8(1):43–68.

Mani, I. and Maybury, M. T. (2001). Automatic summarization. In *Association of Computational Linguistics (Companion Volume)*, page 5.

Mann, W. C. and Thompson, S. A. (1998). Rhetorical structure theory: Toward a functional theory of text organization. *Text – Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281.

Marcu, D. and Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 368–375, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marcu, D. and Gerber, L. (2001). An inquiry into the nature of multidocument abstracts, extracts, and their evaluation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics Conference - Workshop on Automatic Summarization (NAACL 2001)*, Pittsburgh, PA.

Marcu, D. C. (1997). *The rhetorical parsing, summarization, and generation of natural language texts*. PhD thesis, University of Toronto, Toronto, Canada.

Mardia, K., Kent, J., and Bibby, J. (1979). *Multivariate analysis*. Academic Press New York.

Marsi, E., Krahmer, E., Hendrickx, I., and Daelemans, W. (2010). On the limits of sentence compression by deletion. In Krahmer, E. and Theune, M., editors, *Empirical Methods in Natural Language Generation*, volume 5790 of *Lecture Notes in Computer Science*, pages 45–66. Springer-Verlag, Berlin, Heidelberg.

Mateus, M. H. M., Brito, A. M., Duarte, I., Faria, I. H., Frota, S., Matos, G., Oliveira, F., Vigário, M., and Villalva, A. (2003). *Gramática da Língua Portuguesa*. Caminho, 5th edition.

Mckeown, K., Passonneau, R. J., Elson, D. K., Nenkova, A., and Hirschberg, J. (2005). Do summaries help? a task-based evaluation of multi-document summarization. In *Proceedings of the 28th Conference on Research and Development in Information Retrieval*

(SIGIR 2005), pages 210–217, New York, NY, USA. Association for Computing Machinery.

McKeown, K. R., Hatzivassiloglou, V., Barzilay, R., Schiffman, B., Evans, D., and Teufel, S. (2001). Columbia multi-document summarization: Approach and evaluation. In *Proceedings of the Document Understanding Conference (DUC 2001)*.

Medelyan, O. (2007). Computing lexical chains with graph clustering. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Conference: Student Research Workshop (ACL 2007)*, pages 85–90, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing order into texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-04)*, pages 404–411. Association for Computational Linguistics.

Miltsakaki, E., Dinesh, N., Prasad, R., Joshi, A., and Webber, B. (2005). Experiments on sense annotations and sense disambiguation of discourse connectives. In *In Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (TLT 2005)*.

Minel, J.-L., Nugier, S., and Piat, G. (1997). How to appreciate the quality of automatic text summarization? In *Proceedings of the Association for Computation Linguistics and European Chapter of the Association for Computational Linguistics – Workshop on Intelligent Scallable Text Summarization (ACL/EACL 1997)*, pages 25 – 30, Madrid, Spain.

Mitkov, R. (2004). *The Oxford Handbook of Computational Linguistics*. Oxford University Press.

Módolo, M. (2003). SuPor: um ambiente para a exploração de métodos extrativos para a sumarização automática de textos em português. Dissertação de mestrado, Universidade Federal de São Carlos.

Montague, R. (1974). Universal grammar. In Thomason, R. H., editor, *Formal Philosophy: Selected Papers of Richard Montague*, number 222–247. Yale University Press, New Haven, London.

Mori, T. (2005). Japanese question-answering system using a* search and its improvement. *Journal of the ACM Transactions on Asian Language Information Processing (TALIP)*, 4:280–304.

Mori, T., Nozawa, M., and Asada, Y. (2005). Multi-answer-focused multi-document summarization using a question-answering engine. *Journal of the ACM Transactions on Asian Language Information Processing (TALIP)*, 4(3):305–320.

Nenkova, A. and McKeown, K. (2012). A survey of text summarization techniques. In Aggarwal, C. C. and Zhai, C., editors, *Mining Text Data*, pages 43–76. Springer US.

Nenkova, A. and Passonneau, R. J. (2005). Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Conference on Human Language Technology Research and the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2005)*, pages 145–152.

Neto, J. L., Santos, A. D., Santos, R. D., Kaestner, C. A. A., and Freitas, A. A. (2000). Generating text summaries through the relative importance of topics. In *Proceedings of the International Joint Conference Ibero-American Conference on Artificial Intelligence and Brazilian Conference on Advances in Artificial Intelligence (IBERAMIA/SBIA 2000)*, pages 301–309. Springer-Verlag.

Okazaki, N., Matsuo, Y., and Ishizuka, M. (2004). Improving chronological sentence ordering by precedence relation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, pages 750–756, Stroudsburg, PA, USA. Association for Computational Linguistics.

Orăsan, C. (2009). Comparative evaluation of term-weighting methods for automatic summarization. *Journal of Quantitative Linguistics*, 16(1):67–95.

Otterbacher, J. C., Radev, D. R., and Luo, A. (2002). Revisions that improve cohesion in multi-document summaries: a preliminary study. In *Proceedings of the Association of Computational Linguistics Conference – Workshop on Automatic Summarization - Volume 4 (ACL-AS 2002)*, pages 27–36, Stroudsburg, PA, USA. Association for Computational Linguistics.

Over, P., Dang, H., and Harman, D. (2007). DUC in context. *Journal of Information Processing and Management*, 43(6):1506–1520.

Paice, C. D. (1990). Constructing literature abstracts by computer: techniques and prospects. *Information Processing and Management: an International Journal - Special issue on Natural Language Processing and Information Retrieval*, 26(1):171–186.

Pardo, T. A. S. and Rino, L. H. M. (2001). A summary planner based on a three-level discourse model. In *Proceedings of the Natural Language Processing Pacific Rim Symposium (NLPRS 2001)*, pages 533–538, Tokyo, Japan.

Pardo, T. A. S. and Rino, L. H. M. (2004). Descrição do gei – gerador de extratos ideais para o português do brasil. Technical report, São Carlos–SP.

Pardo, T. A. S., Rino, L. H. M., and das Graças V. Nunes, M. (2003). Gistsumm: A summarization tool based on a new extractive method. In *Proceedings of the Encontro para o Processamento Computacional da Língua Portuguesa Escrita e Falada (PROPOR 2003)*, Lecture Notes in Computer Science, pages 210–218. Springer.

Park, J. and Cardie, C. (2012). Improving implicit discourse relation recognition through feature set optimization. In *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2012)*, pages 108–112, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pitler, E. and Nenkova, A. (2009). Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing – Conference Short Papers (ACL-IJCNLP 2009)*, pages 13–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., and Joshi, A. (2008). Easily identifiable discourse relations. In *Proceedings of the Conference on Computational Linguistics – Companion volume: Posters (COLING 2008)*, pages 87–90, Manchester, UK. COLING 2008 Organizing Committee.

Prasad, R., Dinesh, N., Lee, A., Miltsakaki, E., Robaldo, L., Joshi, A., and Webber, B. (2008). The Penn Discourse TreeBank 2.0. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2008)*.

Prasad, R., Miltsakaki, E., Dinesh, N., Lee, A., Joshi, A., Robaldo, L., and Webber, B. (2007). The penn discourse treebank 2.0 annotation manual. Technical Report IRCS-08-01, Institute for Research in Cognitive Science, University of Pennsylvania.

Quinlan, J. R. (1996). Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4(1):77–90.

Radev, D., Otterbacher, J., and Zhang, Z. (2004). Cst bank: A corpus for the study of cross-document structural relationships. In *Proceedings of 4th Language Resources and Evaluation Conference (LREC 2004)*, Lisbon, Portugal.

Radev, D. R. (2000). A common theory of information fusion from multiple text sources step one: cross-document structure. In *Proceedings of the 1st Annual Meeting of the Special Interest Group on Discourse and Dialogue – Volume 1 (SIGDIAL 2000)*, pages 74–83, Stroudsburg, PA, USA. Association for Computational Linguistics.

Radev, D. R., Hovy, E., and McKeown, K. (2002). Introduction to the special issue on summarization. *Journal of Computational Linguistics*, 28(4):399–408.

Radev, D. R., Jing, H., and Budzikowska, M. (2000). Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In *Proceedings of the North American Chapter of the Association for Computational Linguistics and Applied Natural Language Processing Conference – Workshop on Automatic Summarization (NAACL-ANLP-AutoSum 2000)*, volume 4, pages 21–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Radev, D. R. and McKeown, K. R. (1998). Generating natural language summaries from multiple on-line sources. *Journal of Computational Linguistics - Special Issue on Natural Language Generation*, 24(3):469–500.

Ramsay, A. (2004). *The Oxford Handbook of Computational Linguistics*, chapter 6, Discourse. In Mitkov (2004).

Rino, L. H. M. and Módolo, M. (2004). Supor: An environment for as of texts in brazilian portuguese. In *Advances in Natural Language Processing*, volume 3230 of *Lecture Notes in Computer Science*, pages 419–430. Springer Berlin Heidelberg.

Rino, L. H. M. and Pardo, T. A. S. (2003). Temário: Um corpus para a sumarização automática de textos. Technical report, São Carlos – SP.

Rocha, P. and Santos, D. (2000). CETEMPúblico: Um corpus de grandes dimensões de linguagem jornalística portuguesa. In *Proceedings of the 5th Encontro para o Processamento Computacional da Língua Portuguesa Escrita e Falada (PROPOR 2000)*, pages 131–140.

Saggion, H. (2006). Text summarization: Resources and evaluation. In *Proceedings of the Language Resources and Evaluation Conference – Tutorial Notes (LREC 2006)*, Genoa, Italy.

Saggion, H. (2014). Creating summarization systems with SUMMA. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014)*. European Language Resources Association (ELRA).

Schiffman, B., Mani, I., and Concepcion, K. J. (2001). Producing biographical summaries: Combining linguistic knowledge with corpus statistics. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL 2001)*, pages 458–465, Stroudsburg, PA, USA. Association for Computational Linguistics.

Schiffman, B., Nenkova, A., and McKeown, K. (2002). Experiments in multidocument summarization. In *Proceedings of the 2nd Conference on Human Language Technology Research (HTL 2002)*, pages 52–58, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Schütze, H. and Silverstein, C. (1997). Projections for efficient document clustering. In *Proceedings of the 20th Conference on Research and Development in Information Retrieval (SIGIR 1997)*, pages 74–81, New York, NY, USA. Association for Computing Machinery.

Siddharthan, A. (2003). *Syntactic Simplification and Text Cohesion*. PhD thesis, University of Cambridge.

Siddharthan, A., Nenkova, A., and McKeown, K. (2004). Syntactic simplification for improving content selection in multi-document summarization. In *Proceedings of the 20th Conference on Computational Linguistics (COLING 2004)*, page 896, Morristown, NJ, USA. Association for Computational Linguistics.

Silva, J., Branco, A., Castro, S., and Reis, R. (2010). Out-of-the-box robust parsing of Portuguese. In *Proceedings of the 9th Encontro para o Processamento Computacional da Língua Portuguesa Escrita e Falada (PROPOR 2010)*, pages 75–85.

Silvano, M. (2010). *Temporal and Rhetorical Relations: the Semantics of Sentences with Adverbial Subordination in European Portuguese*. PhD thesis, Faculdade de Letras da Universidade do Porto.

Silveira, S. B. and Branco, A. (2012). Enhancing multi-document summaries with sentence simplification. In *Proceedings of the International Conference on Artificial Intelligence (ICAI 2012)*, pages 742–748, Las Vegas, USA.

Steinberger, J. and Ježek, K. (2009). Text summarization: An old challenge and new approaches. In Abraham, A., Hassanien, A.-E., de Carvalho, A. P. L. F., and Snášel, V., editors, *Foundations of Computational, Intelligence - Volume 6*, volume 206 of *Studies in Computational Intelligence*, pages 127–149. Springer Berlin Heidelberg.

Teufel, S. (2001). Task-based evaluation of summary quality: Describing relationships between scientific papers. In *Proceedings of the North American Chapter of the Association for Computational Linguistics – Workshop Automatic Summarization (NAACL 2001)*, pages 12–21.

Teufel, S. and Moens, M. (1999). Argumentative classification of extracted sentences as a first step towards flexible abstracting. In Mani, I. and Maybury, M., editors, *Advances in Automatic Text Summarization*, pages 155–171. MIT Press.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA.

Versley, Y. (2013). Subgraph-based classification of explicit and implicit discourse relations. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, pages 264–275, Potsdam, Germany. Association for Computational Linguistics.

Wang, D., Li, T., Zhu, S., and Ding, C. H. Q. (2008). Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization. In *Proceedings of the 31st Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 307–314. Association for Computing Machinery.

Webber, B. and Joshi, A. (2012). Discourse structure and computation: Past, present and future. In *Proceedings of the Association for Computational Linguistics Conference – Special Workshop on Rediscovering 50 Years of Discoveries (ACL 2012)*, pages 42–54, Jeju Island, Korea. Association for Computational Linguistics.

Wellner, B., Pustejovsky, J., Havasi, C., Rumshisky, A., and Saurí, R. (2006). Classification of discourse coherence relations: an exploratory study using multiple knowledge sources. In *Proceedings of the 7th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL 2006)*, pages 117–125, Stroudsburg, PA, USA. Association for Computational Linguistics.

White, M., Korelsky, T., Cardie, C., Ng, V., Pierce, D., and Wagstaff, K. (2001). Multidocument summarization via information extraction. In *Proceedings of the 1st Conference on Human Language Technology Research (HLT 2001)*, pages 1–7, Morristown, NJ, USA. Association for Computational Linguistics.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco.

Wives, L. K. (2004). *Utilizando Conceitos como descritores de Textos para o processo de identificação de conglomerados (clustering) de documentos*. PhD thesis, Instituto de Informática, UFRGS, Porto Alegre.

Wolf, F. and Gibson, E. (2005). Representing discourse coherence: A corpus-based study. *Journal of Computational Linguistics*, 31(2):249–288.

Wong, K.-F., Wu, M., and Li, W. (2008). Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1 (COLING 2008)*, pages 985–992, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yoshikawa, K., Iida, R., Hirao, T., and Okumura, M. (2012). Sentence compression with semantic role constraints. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics – Volume 2: Short Papers (ACL 2012)*, pages 349–353. The Association for Computer Linguistics.

Zajic, D., Dorr, B. J., Lin, J., and Schwartz, R. (2007). Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing and Management: an International Journal - Special Issue: Summarization*, 43(6):1549–1570.