Universidade de Aveiro
Departamento de
Electrónica, Telecomunicações e Informática
2017

Bruno Manuel
Ferreira Marques

Experiências na Extração de Relações Genéricas em Português

Bruno Manuel
Ferreira Marques

Experiências na Extração de Relações Genéricas para Português

Information Extraction Platform

Tese apresentada à Universidade de Aveiro, para cumprimento dos requisitos necessários para obtenção de grau académico de Mestre em Engenharia de Computadores e Telemática, no âmbito da extração de informação, realizado sob a orientação do Professor Doutor António Joaquim da Silva Teixeira, Professor Associado do Departamento de Eletrónica Telecomunicações e Informática da Universidade de Aveiro, e pelo Professor Doutor Mário Jorge Ferreira Rodrigues, Professor Adjunto da Escola Superior de Tecnologia e Gestão de Águeda da Universidade de Aveiro.

o júri / the juri

presidente / president

Doutor Luís Seabra Lopes
Professor Associado, Universidade de Aveiro

vogais / examiners committee

Doutor Alberto Simões
Professor Adjunto Convidado, Instituto Politécnico do Cávado e do Ave

Doutor António Joaquim da Silva Teixeira (Orientador)
Professor Associado, Universidade de Aveiro

Agradecimentos

Dedico todo este trabalho e percurso académico sobretudo e fundamentalmente à minha tão adorada mãe, Rosinda da Conceição Ferreira Marques, o seu incansável apoio em todos os sentidos à minha pessoa, dedico também ao meu pai Manuel Roque Marques e à minha irmã Sónia Alexandra Ferreira Marques. Agradeço à Universidade de Aveiro, por permitir a existência deste tão grandioso curso, aos meus amigos, aos meus orientadores Professor Doutor António Joaquim da Silva Teixeira e Professor Doutor Mário Jorge Ferreira Rodrigues a persistência nos bons resultados a ajuda e a abertura tanto às ideias por mim expostas, como também ao relacionamento permitido.

Palavras-Chave Informação, Extracção de Informação, OllIE, Freeling, Maltpaser, NLP, Natural Language Processing

Resumo A informação está cada vez mais presente nos dias de hoje e aumenta exponencialmente a cada instante, trazendo por consequência informação não filtrada e como tal informação não necessária ou não objetiva. Para tentar colmatar esse facto foi desenvolvido um projeto que visa conseguir extração de informação na língua portuguesa, através de relações genéricas. Sabendo à partida que essas moldam um padrão que caracteriza uma determinada precisão, o objetivo do sistema que aqui se apresenta assenta na extração de informação de texto que rege uma ligação entre argumentos, com a particularidade de evitar uma pré definição de relações à priori. Os sistemas de Extração de Informação partilham de três fraquezas que não se devem descurar.

1. A extração é normalmente feita com base em verbos.
2. É ignorado o contexto, pelo que os tuplos extraídos não podem ser afirmados como factuais, salvo algumas exceções, como as datas.
3. Existe a dificuldade de Extração de informação (EI) na língua Portuguesa devido ao tagset utilizado, sendo que a língua aceite nestes termos é sobretudo o Inglês, cujo tagset é Penn Treebank[25] e que não é diretamente mapeado para um tagset de língua Portuguesa

Esta dissertação de mestrado tem como objetivo inicial o processamento de texto sob a língua portuguesa e que assenta na extração de informação com relações genéricas em Português e que retratem com alguma objetividade o geral do que se trata o texto.
Os métodos utilizados que interagem entre si, observam e processam o texto com o objetivo de formar uma solução no tema proposto. Todos esses passos serão descritos e dissecados nesta dissertação.
Foram executadas avaliações de versões e testados resultados, nas quais se caracteriza sobretudo a perceção da informação extraída, precisão de modelos de extração tendo como foco relações tipo e não descurando o processamento do output e a forma como seria apresentado ao utilizador. Esperam-se extrações de informação que se complementem e façam sentido entre as mesmas e que para além de relações entre argumentos, nos seja dada informação sobre o texto processado.
O objetivo fora concretizado na medida em que o software proposto não só extrai informação de domínio genérico, como também essa informação se complementa à medida que é extraída, fazendo com que haja uma linha de continuidade na informação, retratando pontos importantes no texto inicial e com isso obtendo resultados positivos no domínio da extração de informação e dando um contributo na área, ainda que tenha limitações bem visíveis e que demoveram alguns dos objetivos propostos inicialmente.

Keywords Information, Extraction, Generic Relations, OllIE, Freeling, Maltparser, NLP, Natural Language Processing

Abstract

Information is increasingly present today and increases exponentially at every moment, bringing unfiltered information as a consequence with non-necessary or non-objective information.
In order to try to overcome this, a project was developed to extract information in Portuguese language through generic relations. Knowing at the outset that these mold a pattern that characterizes a certain precision, the purpose of the system presented here is based on the extraction of text information that governs a link between arguments, with the particularity of avoiding some definition of any relations.
Information Extraction systems share three weaknesses that should not be overlooked.

1. Extraction is usually done based on verbs.
2. The context is ignored, so the extracted tuples can not be stated as factual, except for some exceptions, such as dates.
3. There is a difficulty in Information Extraction (IE) in Portuguese language due to the tagset used, and the language accepted in these terms is mainly English, whose tagset is Penn Treebank[25] and which is not directly mapped to a Portuguese language tagset

This Master's Thesis aims at the processing of text under the Portuguese language and based on the extraction of information with generic relationships in Portuguese and describe, with some objectivity, the general of what the text is about.
The methods used interact with each other, observe and process the text with the objective of forming a solution in the proposed theme. All these steps will be described and dissected in this dissertation.
We performed evaluations of versions and tested results, in which it is mainly characterized the perception of the information extracted, precision of extraction models having focus relations and not neglecting the processing of the output and the way it would be presented to the user. Extractions of information are expected to complement each other and make sense among them and that in addition to relations between arguments, information about the text processed is given to us.
The objective was accomplished insofar as the proposed software not only extracts information from generic domain, but also complements the information as it's extracted, making a continuity line of information, highlighting important points in the initial text and having positive results in the field of information extraction and making a contribution in the area, although it has very visible limitations and that have demolished some of the goals initially proposed.

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

"Se toda a informação é uma poderosa arma, então não é a quantidade que dá o valor, mas sim o acesso no momento certo e na forma mais adequada." Por: Autor desconhecido

1.1 Enquadramento e Motivação

Obter uma informação sobre algo objetivamente especificado numa grande coleção de artigos é um procedimento complexo, moroso e difícil, vivendo na era da informação, cuja tende a crescer dia após dia de forma exponencial, deparamo-nos com a problemática de grandes quantidades de informação, o que dificulta a sua compreensão por parte de um utilizador comum. Para isso nasceu a necessidade de extração de informação que visa extrair automaticamente informações estruturadas de documentos não estruturados e/ou semi-estruturados legíveis por máquina. Na maioria dos casos, esta atividade diz respeito ao processamento de textos de linguagem humana por meio do processamento de linguagem natural e que passa por obtenção de informação com o recurso a modelos de extração que filtram a mesma, o que se mostra muito mais fácil e objetivo, da mesma forma que a procura de uma interação particular sob dois sujeitos será muito mais fácil se as suas relações forem especificadas com antecedência. Alguns exemplos de uso focam sobretudo no resumo de grandes quantidades de dados, atividades recentes no processamento de documentos multimédia como a anotação automática e a extração de conteúdo de imagens / áudio / vídeo podem ser vistas como extração de informações. Devido à dificuldade do problema, as abordagens atuais focam a Extração de Informação (EI) em domínios estritamente restritos, através de uso de ontologia, em que a extração é realizada de acordo a relações mencionadas à priori. Um exemplo é a extração de relatórios de notícias de fusões corporativas, tais como denotado pela relação formal: argumento1, relação, argumento 2.

A extração de informação é um sub-área do processamento de linguagem natural, mais conhecido por Natural Language Processing ou NLP, dedicado exclusivamente à problemática geral de deteção de entidades referidas nos textos de linguagem natural, relações entre elas e eventos nas quais participam [28]. Os textos de linguagem natural podem ter também algum tipo de markup e que consigam ser lidos informaticamente. Como Gai-

zauskas[11] observou, a extração de informação pode ser vista como fontes de informação estruturadas a partir de fontes não estruturadas a qual difere de recuperação de informação, a tarefa de localizar documentos que completem um pesquisa em grandes conjuntos de documentos normalmente executados pelos motores de busca atuais, como Google e Bing, uma vez que o seu objetivo é adquirir informações relevantes que possam posteriormente ser manipuladas conforme necessário, a EI permite, por outro lado, adquirir informações relevantes do conteúdo desses mesmos documentos. Sistemas de extração de informação, numa visão geral, são capazes de fundir informação relacionada encontrada nos documentos, produzindo sumários dos factos reportados em grande escala. As primeiras tarefas da extração visam a concentração em torno da identificação de entidades, como pessoas, nomes de empresas, locais, entre outros e relações entre essas[32]. No estado da arte dos últimos anos, novos sistemas apareceram, concentrando-se em línguas como o Inglês, Sueco, Espanhol e Alemão, que permitem o acesso a dados estruturada ou não-estruturada, aplicando as técnicas de processamento de língua natural. É sabido que existe muito pouco no que diz respeito à EI em Português, ainda que alguns trabalhos existam, estes não são relevantes, sendo que é um foco fundamental no decorrer deste projeto.

O OpenIE [3] conhecido como Open Information Extraction, é visto como um desafio aberto de extração que facilita o domínio da descoberta de relações extraídas de texto que escala para a diversidade e tamanho do corpus. O OpenIE refere-se então à extração de tuplos de relações tipicamente binários, ou seja, que ligam dois argumentos, a partir de texto simples. A diferença central é que o esquema para essas relações não precisa de ser especificado antecipadamente, normalmente o nome da relação é apenas o texto que liga dois argumentos correspondentes à relação de domínio aberto. Como resultado, os sistemas OpenIE tradicionais contam com linguística forte e ajustada ao domínio de interesse, como analisadores de dependência (dependency parsers) e NER (Named Entity Recongnizers). De entre os sistemas OpenIE mais conhecidos podemos encontrar o ReVerb[16], OpenIE[3], StanfordNLP[24] entre outros importantes no decorrer deste trabalho.

Na sequência do trabalho já realizado na área da extração de informação, nos quais se focam aspetos como NLP, Extração de Informação com base em ontologias e relações especificas, entre outros, foi necessário dar continuidade aos projetos anteriores. A proposta para o desenvolvimento de uma ferramenta de software que conseguisse analisar todo um conjunto de dados na língua Portuguesa surgiu no âmbito de preencher a falta de possibilidades existentes neste campo de modo a identificar atributos que resumam a informação presente sobre um dado texto, tendo sobretudo como foco e objetivo a contribuição para o avanço na área, análise e extração de informação na língua Portuguesa.

O tema desta dissertação de Mestrado assenta na problemática da extração de informação em relações genéricas para o Português em dados sob a forma de texto não estruturado, mais propriamente extração de informação em qualquer tipo de texto, sejam notícias, textos, livros, informações sobre produtos, artigos, entre outros e tendo como foco principal a língua Portuguesa.Foi dada especial atenção ao melhoramento de técnicas usadas em aproximações anteriores, cujas extraíam apenas relações pré-definidas e portanto era necessário reduzir essa limitação.

A motivação existente para o desenvolvimento nesta área prende-se com a falta de

soluções para este tema e também com um mercado competitivo que está cada vez mais dependente da informação que existe na Internet, sejam eles consumidores, seja para uso na gestão, seja para resumo de informação de grandes quantidades, pesquisa ou para qualquer outra utilização que se relacione com a procura de informação e respetiva relação. Também as limitações existentes e já abordadas como a não existência de um software que permitisse uma extração limpa, com relações genéricas foram parte de uma motivação que advêm justamente de se conseguir processamento na língua portuguesa.

Um exemplo corrente disso é saber-se à partida que a quantidade de informação existente na Internet é muito grande e extremamente vaga e que essa informação quando bem usada pode ser muito influente nas decisões dos consumidores, estes cujas opiniões contam como uma das mais influentes fontes de decisão. Então, para se ter uma perceção sobre determinados produtos e serviços ter-se-ia que analisar toda essa informação, o que seria extremamente moroso ou até mesmo impossível e usar pequenas amostras não daria a garantia da veracidade do que se estava a extrair. Sendo necessária uma solução que visa aproximar a realização desse trabalho pelo utilizador através de um conjunto de iterações pelo qual a informação é direcionada e tratada. Este processo é uma solução que aponta para um avanço no desenvolvimento da extração de informação para a língua Portuguesa. Mais sobre este tema é apresentado e explicado no decorrer desta dissertação de mestrado.

1.2 Problema(s)

A falta de recursos existentes neste campo, como corpora e sistemas, de modo a identificar atributos que resumam a informação presente em dados sobre um documento que pode ser um produto ou serviço a analisar, representa um problema na área de extração de informação, esta tem várias alternativas existentes e em curso, porém o seu uso está maioritariamente e especificamente confinado ao Inglês, existindo muito pouco na técnica para o Português, muito devido à tagset utilizada, cujo mapeamento da mesma para Português não é de todo direto, sobretudo pela complexidade da língua Portuguesa.

São usadas ferramentas como OpenIE[3], OllIE[35], GATE[10], Reverb[16], este último que é o antecessor do OllIE, um software que obtém alto rendimento ao extrair relações mediadas por substantivos, adjetivos, entre outros e que observa o contexto, que tem uma maior precisão ao incluir informações contextuais do texto analisado.

Estes trabalhos apresentados são os mais conhecidos na área e neste momento representam o estado da arte no que diz respeito à Extração de Informação, usam o tagset Penn TreeBank[25]. Não se encontrou software especifico de Extração de Informação com relação genérica com base no Português, no entanto existem trabalhos que mesmo sendo limitados, não são de software público ou são limitados, pelo que o trabalho apresentado tenta preencher essa lacuna com o objetivo de fornecer algum apoio e avanço direcionado os esforços exclusivamente para a língua Portuguesa de modo a que seja um ponto de partida para que esta problemática seja solucionada, prestando alguma motivação com este trabalho de pesquisa e desenvolvimento. No entanto é de frisar a falta de suporte em geral para Português, assim como a dificuldade inerente à área no que diz respeito aos conhecimentos

necessários para operar com ferramentas deste calibre, ou seja, não foram desenvolvidas para ser "user friendly" dado que apresentam imensa complexidade ao utilizador comum.

1.3 Objectivos

Neste projeto tenta-se responder à falta de meios existentes na área da extração de informação na língua portuguesa e por consequência visa de alguma forma dar continuidade ao trabalho existente já abordado anteriormente. Foi proposto o desenvolvimento de uma ferramenta de software que conseguisse analisar todo um conjunto de dados na língua Portuguesa de modo a identificar atributos e relações que liguem dois argumentos e que resumam a informação. Com este objetivo proposto tenta-se dar um contributo positivo na extração de informação na língua Portuguesa.

Esta dissertação propõe-se então a desenvolver um software capaz de extração de textos em Português tendo como base o sistema de extração OllIE, que, como afirmado, foi desenvolvido para trabalhar com o tagset Penn Treebank, que é utilizado para o Inglês, tendo sido efetuadas todas as adaptações necessárias para que o mesmo extraísse na língua portuguesa, portanto foi necessária a elaboração de um programa que lide com essa diferença de língua e que faz a correspondência entre o tagset Inglês e o tagset Português, assim como os ficheiros necessários para que exista uma extração válida. No decorrer do projeto, gerou-se um pipeline adequado ao sistema de extração, adaptando e desenvolvendo módulos essenciais como o parser morfológico e parser sintático. Por fim, os testes mediados por vários textos sintáticos e corretamente estruturados, validam as nossas escolhas com os resultados obtidos.

1.4 Estrutura da Dissertação

Com este Capítulo 1 foi introduzido o tema no qual se vai focar toda esta dissertação, dando uma breve introdução, enquadramento e motivação.No Capítulo 2 começamos por analisar o trabalho em curso e estado da arte, aqui são abordadas as ferramentas representativas e sobretudo recursos para Português, sendo que no Capítulo 3 entramos numa fase de desenvolvimento de solução propriamente dita, na qual vamos proceder à escolha de ferramentas e apresentar arquiteturas complementadas com diagramas explicativos, passamos depois à modelação do pipeline para software final. No Capítulo 4 serão analisadas essas escolhas e versões de software quanto ao seu uso e solução no presente estado, onde se abordam tópicos como os objetivos de cada versão, assim como a motivação para o desenvolvimento da mesma e posteriormente resultados obtidos em cada uma. Para concluir, no Capítulo 5 será dado o resumo do trabalho efetuado, assim como ideias para o futuro do software e Extração de Informação em geral.

Capítulo 2

Trabalho Relacionado e Estado da Arte

2.1 Introdução

Nesta secção damos a conhecer alguns trabalhos em curso sobre o tema extração de informação, como mostra o capítulo 1 existem alguns, contudo estes são maioritariamente concebidos e focados na língua inglesa com objetivo de identificar e extrair relações específicas entre dois argumentos de um dado texto, pelo formato de argumento1, relação, argumento2. Isto é feito sem um conjunto pré-especificado de relações e sem domínio especifico, passando por um conjunto de ferramentas desenvolvidas e corretamente inseridas em pipeline que têm como fim o tratamento dos dados. Existe uma grande vontade de escalar e desenvolver a extração de informação para um maior conjunto de relações e um maior corpus, sendo que na língua inglesa esse objetivo é aos poucos conseguido devido à quantidade de instituições, como universidades e centros de linguística, que trabalham para que haja esse avanço neste campo, por outro lado, na língua Portuguesa o avanço é muito menor face ao exposto.

Para que se perceba do que se trata na realidade extração de informação, tem de se perceber o que é necessário para que a mesma aconteça. No quotidiano estamos sujeitos a grandes quantidades de informação, sendo que a filtragem das mesmas é um aspeto importante e a considerar e para que com isso se consiga perceber do que se trata essa informação em pouco texto, no fundo é esse o objetivo da extração de informação. Com essa necessidade é fulcral um software que consiga extrair informação de texto em Português com alguma precisão, para que com isso seja possível analisar um conjunto de textos ou documentos locais ou retirados da web, estes podem podem ser notícias escolhidas a partir de jornais online escolhidos aleatoriamente, documentos processados por entidades que estejam disponíveis para uso, ou qualquer outro tipo de dados.

O software proposto, propõe a entrada de documentos ou textos que podem estar sobre qualquer uma das seguintes formas: (1) o texto pode ser texto livre, tirado diretamente de qualquer sitio, desde que possa ser lido computacionalmente. (2) O texto poderá estar sob o formato de XML, em que o mesmo está pré formatado com etiquetas que indicam as entidades

O ponto número um, (1), indica documentos e textos na forma geral, do qual as notícias são um dos exemplos mais específicos e testados, o segundo ponto, (2), aborda o tipo de documento que se pode retirar do site da floresta sintática, a Coleção Dourada do Segundo HAREM, ou seja, os textos além de terem de fazer algum sentido, têm de ter anotadas manualmente as entidades mencionadas, o que elas representam e sobretudo terem uma identificação, sendo que com estas particularidades poderá ser dado como entrada no software e posteriormente analisado. De seguida é apresentado um exemplo do tipo de texto referido no ponto dois (2), com as devidas anotações, retirado do documento mencionado.

```
<P>Era professora, quero dizer ela veio para
<EM ID=‘‘H2192’’ CATEG=‘‘LOCAL’’ TIPO=‘‘HUMANO’’ SUBTIPO=‘‘PAIS’’
COREL=‘‘H2182’’ TIPOREL=‘‘inclui’’>Portugal</EM>
dar aulas. Veio interna para uma casa e no
<EM ID=‘‘H21920’’ CATEG=‘‘TEMPO’’ TIPO=‘‘TEMPO_CALEND’’ SUBTIPO=‘‘DATA’’
COMENT=‘‘DUVIDA_DIRECTIVASTEMPO’’>domingo</EM>
deixou.
</P>
```

2.2 Extração de Informação (EI)

A extração de informação é um poderosa arma e pode ser usada para diversos objetivos, em informática, essa extração é um tipo de recuperação de informações cujo objetivo é extrair automaticamente informações estruturadas por argumentos e sua ligação. As informações estruturadas podem ser, por exemplo, dados categorizados contextualmente e semanticamente bem definidos de documentos não-estruturados legíveis por máquina em um domínio particular. O significado da Extração de Informação é determinado pela quantidade crescente de informação que está disponível na forma não estruturada, isto quer dizer que podemos ter uma melhor extração se podermos transformar em formato relacional. Uma aplicação típica da EI é a procura num conjunto de documentos que é escrito numa linguagem natural e a consequente amostragem das informações extraídas, além disso, a EI é um processo deveras complexo e requer um conjunto de ferramentas que trabalham em linearidade e sintonia de modo a proceder a análise de um determinado texto para a obtenção de uma extração que faça algum sentido ao utilizador. Para esse fim foi necessário um trabalho contínuo de pesquisa e teste, assim como programação objetiva e orientada ao tratamento das palavras, olhado para cada texto como uma frase como um conjunto de palavras contínuas que fazem sentido entre elas e que concluem ou exprimem algo. Esta secção serve para mostrar o que foi aprendido através do olhar atento sobre o que se passa em “pano de fundo” quando se fala em extração de informação. Explicando cada funcionalidade.

Os dados ou documentos a analisar são submetidos como entrada num conjunto de módulos pré programados que terão o papel de os analisar, esta análise atua de modo a que sejam divididos em frases e posteriormente em palavras, encontrando todos os atributos

relacionados com cada uma. Depois dessa análise e tendo todos os atributos que definem uma relação entre as palavras que formam a frase, entra o módulo de extração, esse módulo irá percorrer o texto tendo em conta o anteriormente explicado, ou seja, os atributos correspondentes a cada palavra e frase e proceder à comparação dessas relações com as relações presentes no ficheiro de modelos de extração. Assim só serão captadas as relações existentes que se mostrem positivas de acordo com esse ficheiro de modelos. Segue uma figura que explica o modelo de abordagem usual ao ficheiro de entrada de dados.

Espera-se no aspeto final de extração uma coerência mínima aceitável que defina argumentos e suas relações e que analogamente exprima de forma sucinta a que se deve essa relação, tendo como fundamento o argumento que a origina. Serão mostrados exemplos que comprovam isso mesmo.

2.2.1 Abordagem usual

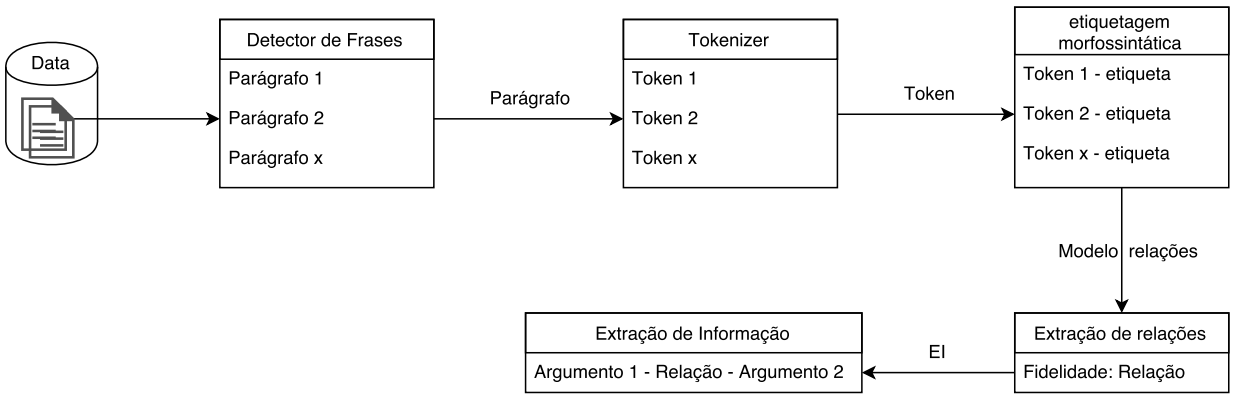

Figura 2.1: Arquitetura da abordagem usual explicado desde a entrada de data, processos e analise textual e triplos extraídos

Exemplo de extração na secção 2.2.2.8.

2.2.2 Sub-tarefas/Módulos

Passamos a falar nesta primeira fase das sub-tarefas típicas que um Extrator de Informação requer para que haja uma extração viável e de seguida será abordado o estado da arte.

O pipeline em termos gerais, começa com uma qualquer entrada de texto tipicamente no formato UTF-8, esses textos podem ser notícias, livros, arquivos contendo informação ou qualquer outro tipo de dados a ser extraído, no fundo pretende-se texto que faça sentido e que possua argumentos, entidades e uma qualquer relação que as junte com algum tipo de significado. Analisemos as sub-tarefas necessárias:

2.2.2.1 Sentence Detector

Os dados de entrada são analisado com o Sentence Detector, este é o modulo responsável de analisar o texto como um todo e dividí-lo por linhas, de modo a formar frases ou parágrafos cujos terminam com um ponto final, o que dá por terminada uma iteração de deteção de frase. Este processo é depois iterado ao longo do texto.

2.2.2.2 Tokenizer

Este que produz uma palavra por cada linha, normalmente é executado depois do sentence detector. A tokenização ocorre no nível da palavra, no entanto, às vezes é difícil definir o que se entende por uma "palavra" e muitas vezes um tokenizer baseia-se em heurísticas simples, por exemplo, a pontuação e o espaço em branco, quebra de linha ou por caracteres de pontuação podem ou não ser incluídas na lista resultante de tokens. Todas as sequências contíguas de caracteres alfabéticos fazem parte de um token, da mesma forma que com números.

2.2.2.3 Análise Morfológica

Dada por terminada esses dois processos de pipeline, o texto passa por uma análise morfológica, que vai fazer o estudo da estrutura e da formação das palavras. A Análise morfológica analisa a classe gramatical dos elementos que formam um enunciado linguístico individualmente, sem que haja ligação entre eles. São classes gramaticais: substantivo, artigo, adjetivo, número, pronome, verbo, advérbio, preposição, conjunção e interjeição.

2.2.2.4 Análise Sintática

A análise sintática por sua vez analisa a função e a ligação de cada elemento que forma um enunciado linguístico, do mesmo modo, a análise morfossintática analisa os elementos do mesmo enunciado linguístico sintática e morfologicamente, o que produz uma saída de informação analisada pronta a entrar num Part-of-Speech Tagger.

2.2.2.5 PoS Tagging

Na linguística do corpus, a análise sintática representa a marcação de POS Tagging, também chamado de Tagging gramatical ou desambiguação de palavras, este é o processo de marcação de uma palavra em um texto (corpus) como correspondente a um determinado papel no discurso e é extremamente importante pois todo o conhecimento de palavras por parte do software vai ter como base a sua definição e o seu contexto. Etiquetador gramatical é um software que aplica a cada palavra que fora tokenizada numa determinada língua a correspondência a uma forma gramatical, isto é, a sua relação com palavras adjacentes e relacionadas numa frase ou parágrafo na identificação de palavras como substantivos, verbos, adjetivos, advérbios, etc. Estes sistemas podem ser baseados em regras ou processos estocásticos, sendo o mais utilizados o algoritmo de E. Brill[6], que opera segundo regras.

2.2.2.6 Chunker

Chunker é uma forma de fazer análise sintática de forma não aprofundada e que por sua vez relaciona as palavras e as suas dependências, a qual gera a árvore de relações que interessa para EI, depois analisada pelo extrator.

2.2.2.7 Named Entity Recognition

O Named Entity Recognition é usado para que seja possível uma caracterização de nomes de pessoas, organizações, locais, expressões temporais e tipo de expressões numéricas, entre outros é necessário a utilização de um NER[1]. No campo do NER existem várias alternativas que são disponibilizadas online, em formato de conteúdo pré programado ou API para programação. Um NER analisa o texto e reconhece as entidades que lá se encontram, sendo que as mais usuais as datas, nomes, organizações e eventos.

O objetivo de um NER num extrator de informação é o facto de poder reconhecer entidades e saber o que é determinada palavra como um nome ou organização, com base nessa informação irá obter-se um novo tag para a mesma, deixando essa de ser um nome próprio, ou adjetivo e passar a ser como foi identificada pelo NER, com isso extrair-se informação contendo essa palavra como o seu novo tag, é portanto o alcance de determinada extração.

2.2.2.8 Extração de entidades e relações

Tendo esses passos de pipeline inicial de tratamento do texto bem definidos, estamos na condição de processo de extração. Este processo é regido por um conjunto de regras que são aplicadas ao texto devidamente analisado, cujo fará sentido, pois sabemos à partida o que representa cada palavra e qual a sua relação e dependências na frase. No entanto convém especificar que existem sistemas de extração que são mediados por uma ontologia, esta é uma denominação e definição formal dos tipos, propriedades e relações entre elas que são denominadas à partida para um domínio particular do discurso. É portanto uma aplicação prática da ontologia filosófica. Sabendo que existem tipo de sistemas que não usam regras de extração e que puramente fazem EI usando um domínio especifico, ou seja, que capta somente o que é pedido, como por exemplo com base no NER em que podemos captar uma data, um acontecimento e um local. É uma boa prática e aproximação à EI, contudo não é o tema base desta dissertação.

No sistema proposto cada frase e consequentemente cada uma das palavras, após a devida análise já explicada, vão ser submetidas a um ficheiro de modelos de extração por parte da máquina de extração de informação, esta que dará a informação do que é ou não legível para fazer parte de uma extração. Estas são as tarefas típicas de um EI, tendo presente esse conceito, podemos avançar para o nosso sistema. Junto se apresentam alguns exemplos de extração de cada um dos módulos dado o seguinte texto:

"Portugal colocou esta quarta-feira 1250 milhões de euros. Este montante e o máximo indicativo, em Bilhetes de Tesouro a três e 11 meses a taxas de juros médias. Inferiores às dos anteriores leilões comparáveis."

- Sentence Detector:
  - Portugal colocou esta quarta-feira 1250 milhões de euros.
  - Este montante e o máximo indicativo, em Bilhetes de Tesouro a três e 11 meses a taxas de juros médias.
  - Inferiores às dos anteriores leilões comparáveis.
- Tokenizer:
  - Portugal
  - colocou
  - esta
  - quarta-feira
  - 1250
  - milhões
  - de
  - euros. ...
- Analise Morfológica:
  - Portugal portugal AQ0CS00 1
  - colocou colocar VMIS3S0 1
  - esta este DD0FS0 0.877868 este PD0FS00 0.122132
  - quarta-feira quarta-feira NCFS000 1
  - 250 1250 NCFP000 0.836712 1250 RG 0.0971691 1250 NCMS000 0.0333976 1250 NCMP000 0.032721
  - milhões milhão NCCP000 0.649997 milhão NCMP000 0.350003
  - de de SP 1
  - euros euro NCMP000 1
- Analise Sintática e PoS Tagging:

```
Portugal_prop colocou_v-fin esta_pron-det quarta-feira_n 1250_num milhóes_n
    de_prp euros._n Este_pron-det montante_n é_conj-c o_art máximo_n
    indicativo,_adj em_prp Bilhetes_prop de_prp Tesouro_n a_prp trẽs_num
    e_conj-c 11_num meses_n a_art taxas_n de_prp juros_n médias._adj
    Inferiores_prop ás_adv dos_n anteriores_adj leilóes_n comparáveis._prp
```

- Chunker:

```
S_[
  s-adj_[
    +s-a-ms_[
      +a-ms_[
        +(Portugal portugal AQ0CS00)
      ]
    ]
  ]
  grup-verb_[
    +verb_[
      +(colocou colocar VMIS3S0)
    ]
  ]
  sn_[
    espec-fs_[
      +dem-fs_[
        +(esta este DD0FS0)
      ]
    ]
    +grup-nom-fs_[
      +n-fs_[
        +(quarta-feira quarta-feira NCFS000)
      ]
    ]
  ]
  sadv_[
    +(1250 1250 RG)
  ]
  sn_[
    +grup-nom-mp_[
      +n-mp_[
        +(milhoes milhao NCCP000)
      ]
    ]
  ]
  sp-de_[
    +(de de SP)
    sn_[
      +grup-nom-mp_[
        +n-mp_[
          +(euros euro NCMP000)
        ]
        n-fs_[
          +(. . NCFS000)
        ]
```

```
      ]
    ]
  ]
]
...
```

2.3 OpenIE

Muitos dos sistemas de EI extraem apenas relações pré-definidas. Tentando fazer face a esta limitação, surgiram sistemas que têm por objetivo extrair relações genéricas não limitadas a um conjunto previamente definido. Estes sistemas designam-se em Inglês por Open-IE, Extração de informação aberta ou genérica, que como o próprio nome indica, se refere a triplos de relações de domínio aberto, representando um sujeito, uma relação e o objeto da relação. Isso é útil para tarefas de extração de relação onde há dados de treino limitados.

Então o OpenIE refere-se à extração de tuplos, normalmente relações binárias de texto livre com a adição de relação entre esses, o que forma por fim um triplo de extração. A diferença central está no esquema utilizado para estas relações, que não necessitam de ser especificadas. Alguns dos sistemas mais conhecidos que trabalham com OpenIE são o OllIE e ReVerb, que serão tratados aqui.

2.4 Ferramentas

O estado da arte em extração de informação e Natural Language Processing ainda está longe de ser capaz de construir representações de propósito geral com significado a partir de texto sem restrições, portanto será mais vantajoso se focarmos os esforços num conjunto limitado de questões e relações de entidade.

Para a realização deste projeto de mestrado, teve de ser feita uma análise ao estado da arte do momento relativo ao tema, ou seja, análise e processamento de texto. Para que se consiga abordar o campo do tagging de Part-of-Speech, Sentece Detection e consequente divisão, Tokenize de cada palavra e reconhecimento de entidades e como tal entrar na área da extração da informação. Esses são textos que falam sobre determinado tema, acontecimento ou produto, que serão analisados. A pesquisa sobre o estado da arte prende-se então com produtos, sistemas e API existentes com base em formular uma opinião sólida sobre os mesmos e por consequência escolher o melhor caminho para que seja feito um avanço na extração e análise de informação relativamente à língua Portuguesa, que como já foi referido é um dos grandes problemas no campo da extração de informação.

Alguns trabalhos que se seguem estão presentes no campo de ação do tema, esses são módulos que foram analisados com vista a preencher algumas lacunas existentes e que são necessários para o desenvolvimento deste projeto. Para além da caracterização do estado da arte em OpenIE, tema da próxima secção, a criação de um pipeline de processamento

adequado obrigou à identificação e seleção de ferramentas e recursos para as várias subtarefas, em especial as de Part-of-Speech Tagging, Parsing, Named Entity Recognition, Tokenizer, Chunker e Dependencies.

Nesta secção se apresenta alguma informação sobre as mais relevantes assim como os principais módulos de software requeridos para que seja feito um processamento ao input dado, com objetivo de análise, sendo estas algumas ferramentas que eram tidas como partes para formulação de solução.

2.4.1 Ferramentas genéricas

Após uma intensa pesquisa sobre ferramentas que facilitassem a criação do pipeline, procurando sobretudo suporte ao Português e capacidade de adaptação, resultaram os softwares que se apresentam de seguida.

FreeLing

O FreeLing[31], é um ferramenta desenhada com a funcionalidade de ser usada como biblioteca externa, orientada para programadores para qualquer aplicação que necessite do seu serviço, é escrita em C++ e está disponível para a comunidade. Fornece uma série de funcionalidades para variadas línguas, entre as quais se encontra o Português. Algumas das funcionalidades mais importantes para o desenvolvimento e pesquisa no nosso trabalho foram:

- Tokenização de texto
- Separação de frases
- Análise morfológica
- Deteção de entidades nomeadas
- Reconhecimento de entidades nomeadas
- Etiquetagem de PoS

O dicionário em Português que auxilia o FreeLing contém mais de 909.000 formas correspondentes a 105.000 combinações em Part of Speech e utiliza o tagset EAGLES[36].

Este software não é uma ferramenta de análise de texto orientada ao utilizador comum, ou seja, não é projetado para ser "user-friendly", ou ter uma unidade gráfica e ser simples e bonito, mas sim um software que ajuda programadores a desenvolverem os seus próprios projetos.

Os resultados esperados do FreeLing são a análise linguística em estruturas de dados. Cada aplicação final criada pode aceder a esses dados e processá-los de acordo com o especificado. O analisador fornecido como teste, funciona de acordo com um conjunto pré definido por um pipeline dado pelo utilizador, com as ferramentas que este necessita usar para processamento, poderão cobrir a maioria dos recursos do FreeLing.

Stanford CoreNLP

O Stanford CoreNLP [24] que disponibiliza os recursos presentes no OpenNLP [30], este é uma biblioteca de aprendizagem que usa por base machine learning para processamento natural desenvolvido pela Apache, este suporta tarefas como tokenization, sentence segmentation, Part of Speech tagging, named entity extraction, sendo este último o referido neste ponto, não descurando o uso dos anteriores que foram também necessários. Este pacote é implementado e otimizado para um uso programático em JAVA.

O CoreNLP é projetado para ser altamente flexível e extensível, sendo possível habilitar ou desabilitar opções de funcionamento. Integra portanto muitos blocos de construção fundamentais de compreensão de texto de alto nível e de domínio especifico. É um reconhecedor em estado da arte de entre os inúmeros existentes, foram apenas escolhidos os mais representativos e que estão mais associados ao tipo de aproximação que aqui fazemos.

Essas tarefas são geralmente necessárias para criar serviços de processamento de texto mais avançados. O CoreNLP também incluí entropia máxima e machine learning baseada em perceptrons.

Como referido o CoreNLP tem várias ferramentas importantes, como por exemplo o Stanford Parser, este é um parser estatístico para linguagem natural que trabalha a estrutura gramatical, como a relação entre nós de uma frase. Os analisadores probabilísticos usam o conhecimento da linguagem adquirida a partir de exemplos anotados para tentar produzir a análise mais provável. Um analisador probabilístico lexicalizado implementa um modelo com estrutura de frase PCFG[9], que separa e analisa dependências lexicais, cujas preferências são combinadas por inferência usando um algoritmo A*. Estes analisadores estatísticos ainda cometem alguns erros, mas no geral funcionam muito bem, sendo que esta ferramenta foi um dos maiores avanços no processamento de linguagem natural.
Também o Stanford Log-linear Part-of-Speech Tagger é disponibilizado, este Part of Speech tagger é um software que analisa um texto fornecido em qualquer linguagem compatível com o CoreNLP e fornece um token a cada palavra e assinala-a como nome, verbo, adjetivo, ou outros.

Por último temos o Stanford Open Information Extraction, ou OpenIE. A problemática em torno deste assunto é o recurso à língua utilizada, que é o Inglês, o que para a nossa extração de informação é um ponto critico, como será analisado e explicado.

NLTK

O NLTK[4], Natural Language Toolkit, é uma plataforma em Python desenhada para trabalhar com a língua humana. É uma API que tem uma fácil interface adaptada para qualquer utilizador, fornece processamento de texto, classificação, tokenização, tagging, parsing, entre outros. Tem uma grande comunidade e uma documentação programática bastante útil.

Polyglot

O Polyglot[2] é uma outra ferramenta em Python igualmente poderosa, que permite deteção de linguagem, Tokenization, Part-of-Speech, Named Entity Recognition, Morphological Analyis e Sentiment. Tem uma boa documentação que permite que seja colocada em funcionamento com alguma facilidade. Suporta 40 línguas das quais o Português, é fácil de operar e contém modelos treinados para cada uma dessas línguas. É portanto completo e extremamente usável, tendo sido testada.

Gate

O Gate[10] é um software antigo de código livre, que conta com mais que 15 anos e que permanece ativo e usável para todos os tipos de tarefas computacionais que envolvam linguagem humana. O GATE utiliza o ANNIE, A Nearly-New Information Extraction System, que como o próprio nome indica é um sistema extração de informação, contudo é um sistema com foco empresarial e funciona de acordo com um conjunto gráfico de escolhas no corpus dado como entrada, funciona para o Inglês e não é possível de ser adaptado, no fundo é um set de aplicações pré definidas em formato de pipeline iterado pelo utilizador. Não é portanto o foco para este tema de dissertação.

MaltParser

Não permitindo o tempo disponível uma busca aprofundada de soluções para análise sintática, manteve-se a seleção de trabalhos anteriores tendo por co-autores os orientadores desta Dissertação, concretamente o MaltParser, que é um gerador de analisador de dados para análises de dependências, dado um determinado treebank em formato de dependências e pode ser usado para induzir um analisador para o idioma do treebank, este suporta vários algoritmos de análise e algoritmos de aprendizagem, consistindo em combinações arbitrárias de recursos lexicais, recursos de part-of-speech e recursos de dependências.

Para suporte de algoritmos de parsing, o maltParser baseia-se na abordagem de transição para a análise de dependências, o que significa que um analisador consiste num sistema de transição para derivar árvores de dependências, juntamente com um classificador para prever deterministicamente a próxima transição, dada uma representação de recurso da configuração atual do analisador(Nivre, 2008). Para obter dados de treino para o classificador é usado um esquema para reconstruir uma sequência de transição válida para cada estrutura de dependência no conjunto de treino. Um sistema de transição juntamente com esse esquema define um algoritmo de análise. O MaltParser embora vá na versão 1.8, optou-se pelo uso da versão 1.4.1 devido a compatibilidades de sistema, este está equipado com nove algoritmos de análise diferentes, que diferem não apenas em termos de transições e esquemas, mas também a relação à classe de estruturas de dependência que estes podem manipular. A tabela apresentada lista os valores das opções para o algoritmo de análise de opções para os nove algoritmos, juntamente com informações sobre a classe de estruturas de dependência que cobrem e os ponteiros para a literatura.

parsing algorithm (–a)	Structures	References
nivreeager	Projective	Nivre (2003), Nivre (2008)
nivrestandard	Projective	Nivre (2003), Nivre (2008)
covproj	Projective	Nivre (2003), Nivre (2008)
covnonproj	Non-Projective	Nivre (2006), Nivre (2007), Nivre (2008)
stackproj	Projective	Nivre (2009)
stackeager	Non-Projective	Nivre (2009)
stacklazy	Non-Projective	Nivre et al. (2009)
planar	Planar	GomezRodriguez and Nivre (2010)
2planar	2Planar	GomezRodriguez and Nivre (2010)

Tabela 2.1: Algoritmos de parsing

O algoritmo usado pelo nosso software é o que se apresenta, Arc-Eager e com a configuração de parsing de NivreEager e estrutura projetiva.

```
Transition system  : Arc-Eager
Parser configuration : Nivre with allow_root=true, allow_reduce=false and
    enforce_tree=false
Data Format        : /test/conllx.xml
```

Devido ao facto do tempo requerido para que se faça uma aproximação ao processamento da língua com recurso a documentos ConLL, e que depende sobretudo de anotações feitas à mão sob os documentos a analisar em formato lexical ou gramatical, foi formado o MaltParser, cujo contém uma consistência de precisão de 80% a 90%, dando apenas 5% de probabilidade de erro e que soluciona esse problema. Enquanto um gerador de analisador tradicional constrói um analisador dado uma gramática, um gerador de parser gerado por dados constrói um analisador dado um treebank.

O MaltPaser implementa a dependência indutiva de parsing, onde a análise sintática de uma frase equivale à derivação de uma estrutura de dependência, e onde a aprendizagem por machine learning indutiva é usada para orientar o analisador em pontos de escolha não deterministas. Assim sendo, o MaltParser foi projetado para oferecer flexibilidade máxima na forma como os componentes podem ser variados independentemente uns dos outros.

Também por possuir uma JAVA API foi extremamente útil para dar uma continuidade a este projeto, pois permitiu que existisse o parsing dado um Treebank da floresta sintática, anotado previamente e que nos possibilita a anotação correta da língua.

Tanto o FreeLing, OllIE, MaltParser e CoreNLP foram deveras importantes para o avanço neste projeto, assim como na linguística, extração de informação, análise e processamento de estruturas de dados em Português, portanto será dada especial atenção a estes no decorrer desta dissertação.

2.4.2 Reconhecedores de Entidades

Passemos ao tipo de reconhecedores existentes, como falado existe no projeto OpenNLP cujo objetivo é criar um toolkit bom o suficiente para as tarefas acima mencionadas, adicionalmente fornecendo um grande número de modelos pré-construídos para uma variedade de idiomas, sendo que falha no Português, no que diz respeito ao NER.

Outro existente é o Paramopama[21], que é um corpus para um NER de Brasileiro - Português, conseguindo detetar classes como Pessoa, Localização, Organização e Tempo e cujo utiliza a coleção dourada do segundo HAREM[8], que é uma avaliação conjunta na área do reconhecimento de entidades mencionadas em português. Muito simplificadamente é uma iniciativa que pretende avaliar o sucesso na identificação e consequente classificação automática dos nomes próprios na língua portuguesa.

Outras soluções de Named Entity Recognition foram testadas, programadas e implementadas, como o Stanford CoreNLP[24],

baseado em deep learning e ferramentas rule-based.

Contudo no nosso esquema de projeto não estamos a dar uso a um Named Entity Recognizer propriamente dito, podendo sim fazer uso do mesmo depois de extraídas as informações, para que se tenha presente nomes, organizações, espaços temporais, entre outros, que de alguma forma podem ajudar o utilizador a formar uma melhor opinião sobre uma determinada extração. Para isso foi desenvolvido um programa em Python que caracteriza a informação extraída com auxilio do analisador disponibilizado no pacote FreeLing.

2.4.3 Comentário

Todos estes sistemas, API e soluções, ou parte delas em estado da arte nesta área, estão especificados na maioria para a língua inglesa, esta que é a mais usada neste campo e que difere em muitos aspetos da língua Portuguesa, que está a ser trabalhada nesta dissertação. É portanto um ponto fundamental o desenvolvimento para Português.

2.5 Ferramentas e Recursos para Português

Nesta secção se abordam os recursos existentes para a língua portuguesa no campo de ação da extração de informação, no qual a sua existência é extremamente vaga, daí também uma das motivações para que se avançasse neste projeto.

A Linguateca[33] foi uma organização virtual constituída por quatro pólos localizados em centros de investigação de renome e com experiência em processamento do português, com o objetivo de integrar e potenciar o espírito e a filosofia do projeto inicial, neste caso o Projeto do Processamento Computacional do Português, nos centros que acolhem os pólos, através de uma colaboração prática, desenvolvendo serviços e disseminando os recursos já existentes, contudo, apenas os seus recursos estão disponíveis, pelo que já não está em atividade.

A Linguateca dedica-se a três objetivos fundamentais: (1) - divulgação e catalogação do processamento computacional do português na rede; (2) - disponibilização, melhoria e criação de recursos; (3) - avaliação da área, através da organização de avaliações conjuntas.

Também a solução de problemas como a falta de recursos que possam servir de base ao desenvolvimento de aplicações e ao próprio estudo da língua portuguesa e a falta de métodos e de métricas de avaliação para comparar sistemas e para avaliar o progresso na área, são metas a alcançar. Sendo este o único recurso com alguma estrutura e base para o nosso projeto e cujo toma partido destas soluções disponibilizadas pela Linguateca.

Na organização da Linguateca está inserido o BOSQUE[17], este é um subconjunto da Floresta Virgem revisto por linguistas e com opções de anotação, é portanto uma floresta integralmente revista por linguistas. É composto por 9.368 frases retiradas os primeiros 1000 extractos, aproximadamente, das corpora CETENFolha e CETEMPúblico. Desde 2007 o Bosque passa por um novo processo de revisão, em que foram corrigidas algumas pequenas inconsistências e acrescentadas novas etiquetas. Este é o corpus mais correto da Floresta e por isso o mais aconselhado, como tal foi usado o seu conteúdo que é disponibilizado sob as duas formas já mencionadas a cima, sendo estas o CETENFolha e CETEMPúblico.

O CETEMPúblico ou Corpus de Extratos de Textos Eletrónicos MCT/Público é um corpus de aproximadamente 180 milhões de palavras em português europeu, criado pelo projeto de Processamento computacional do português, sendo este o projeto que deu origem à Linguateca.

O CETENFolha ou Corpus de Extratos de Textos Eletrónicos NILC/Folha de S. Paulo, é um corpus de cerca de 24 milhões de palavras em português brasileiro, criado pelo projeto Processamento computacional do português, com base nos textos do jornal Folha de S. Paulo que fazem parte do corpus NILC/São Carlos. Porém apresenta alguns problemas devido a deficiências no texto original a partir do qual foi criado o corpus.
(1) - Há alguns erros ortográficos no texto
(2) - Em alguns casos não há um espaço entre duas palavras
(3) - Alguns caracteres acentuados desapareceram
(4) - Em alguns casos, existe um sinal de maior (>) em vez do parêntesis reto

Como seria de esperar e por razões óbvias, foi escolhido o CETEMPúblico para nos auxiliar neste percurso. Este ficheiro foi de extrema relevância pelo que foi usado para ser dado como entrada no software Maltpaser já explicado e por consequência formar um ficheiro requerido pelo OllIE para que o português fosse devidamente etiquetado com o tag-set da floresta sintática, este ficheiro tem a extensão ".mco" que é o modelo de parsing que resultou de treinar o MaltParser com o bosque e será abordado ao logo desta dissertação.

Ainda assim, convém especificar a falta de recursos a este nível, pelo que se não fosse com o auxilio dos ficheiros disponibilizados pelo bosque, a extração de informação para o português seria uma impossibilidade.

A extrema falta de recursos a este nível não só elimina o avanço da área como também desinteressa e anula futuras implementações. Contudo organizações como a Linguateca fazem os possíveis para que a área não desapareça e se continue a investir na língua por-

tuguesa como foco informático, neste caso para extração de informação.

2.5.0.1 Análise morfossintática e sintática para Português

O estado da arte em parsers para português é vago, a inexistência de auxiliares de parsing que nos pudesses ajudar a concretizar a visão de extração de informação para a língua portuguesa estava cada vez mais longe e de certo modo mais difícil, pois um dos passos fundamentais é o reconhecimentos das palavras e o que estas representam, sendo que palavras desconhecidas não são base para ligações, são vistos como pontos isolados que não se ligam de modo nenhum. Após alguma pesquisa foi encontrado e analisado um software desenvolvido pela Universidade de Lisboa, LX-Center[5], cujo possui algumas funcionalidades adaptadas para a língua Portuguesa no que diz respeito aos módulos necessários.

O LX-Tagger para Part-of-Speech em Português, atribui uma única marca morfossintática de um conjunto de tags pré definido anexando-o a cada token, é um serviço web disponibilizado no website do LX-center.

Este Software utiliza como recurso o software MXPOST[13] que contém mais de 600.000 tokens com avaliação cruzada dez vezes e com precisão de 96.24 por cento. Ainda que nos ajude a perceber o funcionamento de um Part-of-Speech, foi o primeiro contacto com PoS Tagging, contudo não encaixava no nosso projeto por não ser disponibilizado em código aberto para pesquisa e adaptações.

Este projeto além do mencionado, contém outras ferramentas como LX-Tokenizer, que segmenta o texto em tokens lexicais relevantes usando os espaços em branco como separadores de cada um desses tokens e que também manipula cadeias ambíguas, e como o LX-NER que é um Named Entity Recognizer que reconhece nomes, organizações, eventos, localizações e outros.
Estas são boas ferramentas que poderiam ter sido usadas no decorrer deste tema, e que de alguma forma iriam servir para o mais rápido avanço nos resultados desta tese, contudo, este recurso não foi utilizado por falta de suporte do mesmo, tendo por isso sido abandonado.

Outra ferramenta que foi testada nesta fase foi o TreeTagger[34], este é um Part-of-Speech Tagger que possui informação sobre o lemma e suporte para muitas línguas, incluindo o Português. É uma ferramenta forte, que tal como as ferramentas do LX-center, foi um ponto crucial para o teste de PoS. A base deste consiste em dois programas principais, um treinador de tags, que é usado para criar um arquivo de parâmetros léxicos e um corpus. O treetagger espera um ficheiro de texto e anota o texto com tags Part of Speech. Tem suporte para Windows e possuí uma Graphical User Interface disponibilizada para mais fácil interpretação dos dados.

No mesmo ramo também o TurboParser[27] foi testado, está implementado em C++ e funciona como Pos Tagging e de seguida faz o Parsing, este segundo ponto bastante importante, pois era necessária uma ferramenta que executasse o parsing para dar continuidade à pesquisa. Não foi testado em profundidade, porém é uma ferramenta com boas referências.

Para finalizar o tema do Parser, como referido é um ponto fulcral para o desenvolvi-

mento deste projeto, a necessidade de ter um software de parsing poderoso que se adaptasse à estrutura de banco de dados e fosse de alguma forma configurável em termos mais complexos como o tipo de algoritmo a implementar, tipo de estrutura e modelos, foi de certo modo complicado, contudo e após exaustiva pesquisa chegou-se à solução MaltParser[29], esta solução mostrou ser a melhor devido à forte otimização e configuração de acordo com o objetivo pretendido, assim como a sua documentação.

2.6 OpenIE - Sistemas representativos

Nos sistemas que representam o estado da arte em EI temos o que para nós vai ser o mais importante, o OllIE[35], Open Information Extraction Software, que é um software que identifica e extrai relações binárias de um qualquer texto em Inglês e que está desenhado para extração de informação onde não é requerido qualquer vocabulário pré-especificado, identificando relações e argumentos associados em frases arbitrárias, onde as relações objetivo não podem ser identificadas e a velocidade de processamento é importante.

O ReVerb[16] é um software que identifica e extrai automaticamente relacionamentos binários de frases em Inglês. Este foi projetado para a extração de informações na Web, onde as relações de destino não podem ser especificadas com antecedência e a velocidade processamento de informação é importante. O ReVerb é o antecessor do OllIE, que foi um software extremamente importante para este trabalho, como descrito de seguida.

O OllIE é a segunda geração de extração de informação, cujo vem do estado da arte de softwares como o ReVerb[16] e WOE, que como o OllIE, identifica e extrai relações binárias de um texto formatado na língua Inglesa, no entanto estes dois softwares especificados têm duas falhas que foram tidas em conta e solucionadas no OllIE,

(1) ambos extraem somente relações que estão medidas por verbos e

(2) ambos ignoram o contexto assim como a extração de tuplos que não são assertivos com os factos.

Outra das particularidades do OllIE face ao ReVerb está no facto do ReVerb operar em sequências planas de tokens, enquanto o OllIE trabalha com representação em árvore, fazendo uso da compressão das dependências de Stanford, isto permite que se consiga capturar expressões que o ReVerb não conseguiria, como as relações de longo alcance.

Nesta secção iremos passar pela apresentação do OllIE como solução ideal a implementar nos módulos que formam o pipeline do nosso software, este por ser substancialmente melhorado face aos outros softwares de extração de informação, apresentando factos e limitações. Inicialmente o OllIE alcança um rendimento elevado através das extrações mediadas por substantivos, adjetivos e muito mais, também captura o contexto que modifica uma relação binária, o que faz desta ferramenta uma excelente escolha para o nosso trabalho, obtendo 2.7 vezes mais precisão comparativamente ao ReVerb e WOE.

Enquanto os tradicionais extratores de informação estão focados em identificar e extrair relações de interesse especificados, existe uma grande vontade de aumentar e escalar numa maior conjunto de relações e numa maior corpus No entanto o requisito de ter relações

de interesse pré-especificadas constam com um obstáculo significante, como por exemplo eventos importantes. O estado da arte em OpenIE como o ReVerb aponta para tais cenários, sofrendo de dois pontos chave como o subconjunto limitado de frases para expressar relações e o facto de extrair somente relações mediadas por verbos. Isto deixa passar informações importantes mediadas por outras entidades sintáticas como substantivos, adjetivos assim como uma ampla gama de estruturas verbais [15].
O OllIE tenta colmatar esses problemas dos extratores de informação anteriores, expandido a visão sintática de relações da frase, cobrindo um maior número de expressões de relação e expandindo a representação OpenIE para permitir informações de contexto. As extrações proporcionadas pelo OllIE obtêm uma maior precisão e um comportamento superior face aos sistemas existentes.

A extração de relações feita pelo OllIE pode ser expressada por uma arquitetura que aplica extração binária de padrões, como mostra a figura 2.2, este sistema começa com os tuplos semente que vêm do ReVerb, estes que são usados para construir o conjunto para treino do bootstrap e aprendizagem de templates de padrão aberto. No fim, são aplicados a cada frase para extração.

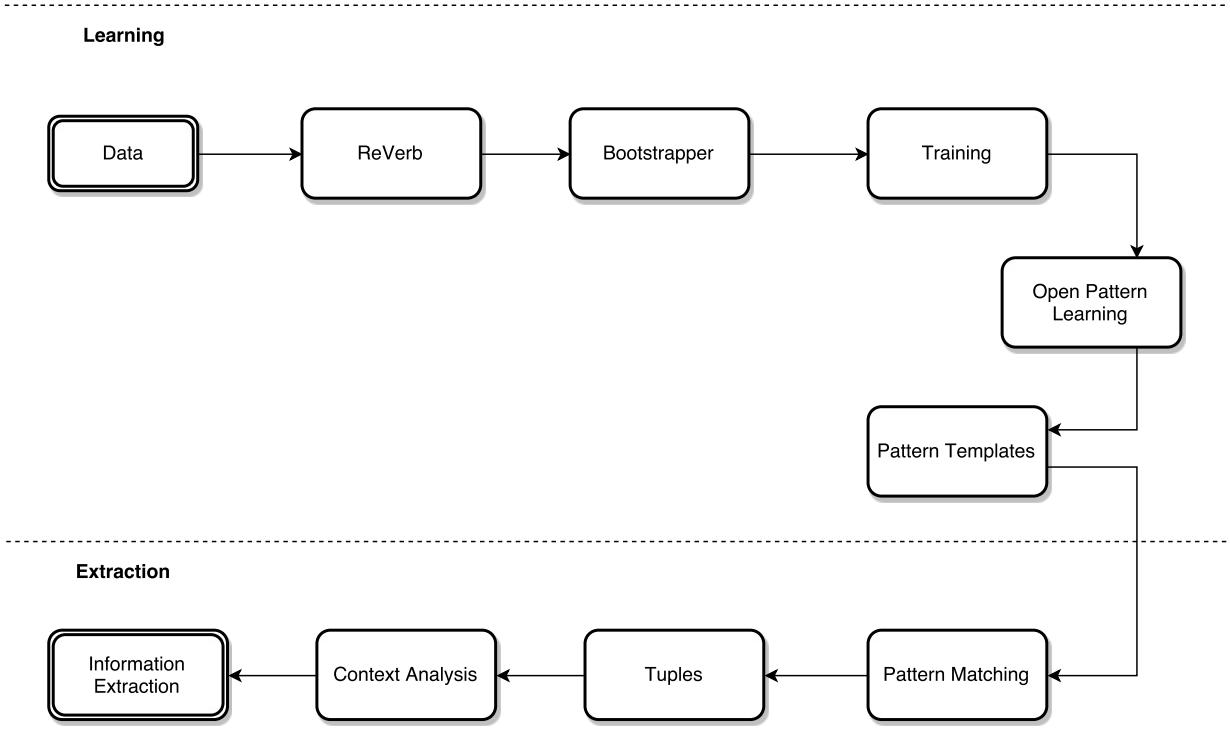

Figura 2.2: Arquitetura do sistema OllIE explicado desde a entrada de tuplos semente, padrão de modelos de extração e por fim a informação extraída

Primeiro usa um conjunto de sementes de alta precisão do ReVerb para bootstrap, em

segundo ele aprende modelos padrão através um conjunto para treino e por último o OllIE aplica esses modelos no momento da extração, esta secção descreve essas três etapas em algum detalhe. Finalmente é analisado o contexto em torno do tuplo para adicionar a informação, também descrito como enabler ou despoletador assim como uma função de confiança.

Para que o objetivo de gerar automaticamente um grande conjunto de treino, que encapsula as múltiplas maneiras de se ter uma informação expressada num dado texto, terá que ser construido o conjunto bootstrap. O fator chave é que quase todas as relações podem ser expressadas via ReVerb, através de expressões baseadas em verbos, então frases bootstrapped baseadas em tuplos ReVerb irão muito provavelmente fazer com que sejam capturadas todas as expressões de relação.

A construção do bootstrap passa por 110.000 tuplos semente que contêm restrições que reduzem a ambiguidade, enquanto continuam a abranger uma vasta gama de relações. Para reduzir os erros bootstrap são aplicadas restrições de dependência, ou seja, só são permitidas frases onde o conteúdo pode ser relacionado entre si, num caminho de dependências inferior a quatro palavras, para que exista fundamento na extração e se diminuam esses erros. Essa restrição é implementada com o uso do subconjunto das palavras que são cabeçalhos na árvore de análise, ou parsed tree. Nesta implementação é usado o Malt Dependecy Parser[29] para o parsing de dependências, pois é rápido, eficaz e facilmente adaptável a grandes quantidades de corpus de frases. É usado para pós-processamento o algoritmo Stanford CCprocessed[26], que é o algoritmo por omissão para dependências da Stanford, este compacta a estrutura analisada para mais fácil extração. Esta forma de bootstrapping é a mais recomendável para o que a extração de informação em formato aberto precisa, uma vez que as relações entre as palavras e as sementes coincide, pode-se aprender modelos de padrão geral que se podem aplicar a outras relações.

Capítulo 3

Framework Genérica para Open IE

3.1 Introdução

Neste Capítulo serão abordadas todas as ferramentas destacadas e testadas para fazer parte da solução para extração de informação na língua portuguesa. Temáticas como a arquitetura e correspondentes diagramas, seleção de ferramentas e posteriores resultados de cada aproximação são explicadas, fazendo assim uso do melhor e mais adequado para composição do nosso software e relativos detalhes de implementação. Também se explica o software FreeLing, qual o seu impacto na continuidade do projeto e posterior relação com o software exposto.

3.2 Arquitectura

Antes de iniciar o trabalho propriamente dito e após os testes e escolhas de ferramentas programáveis que preenchessem os requisitos necessários, foi óbvia a necessidade de discutir uma arquitetura viável que conseguisse juntar os módulos de acordo com um pipeline de extração de informação, usando frameworks como JAVA, Python e C++, na sua essência.

O software inicial passa pela congregação de um conjuntos de documentos dos quais se pretende proceder à extração, esses documentos vão ser submetidos a um conjunto de módulos que os irá processar e tratar, conforme exemplifica o diagrama 3.1

O diagrama 3.1 mostra o pipeline que caracteriza todo o processo levado a cabo pelo software de extração nesta fase, este representa uma arquitetura da qual já faz parte o OllIE e portanto os seguintes representam todo o processo até aqui. Este software é composto pela entrada de ficheiro contendo notícias referentes a empresas e economia para uma fase de teste, mas que pode ser qualquer outro tipo de documento, contendo qualquer outro tipo de informação. Estas passam pelo conjunto de módulos de tratamento formado a partir da API OpenNLP, OllIE com a utilização do MaltParser. Mais sobre os módulos que formam o pipeline, utilizados no presente diagrama, na seguinte secção.

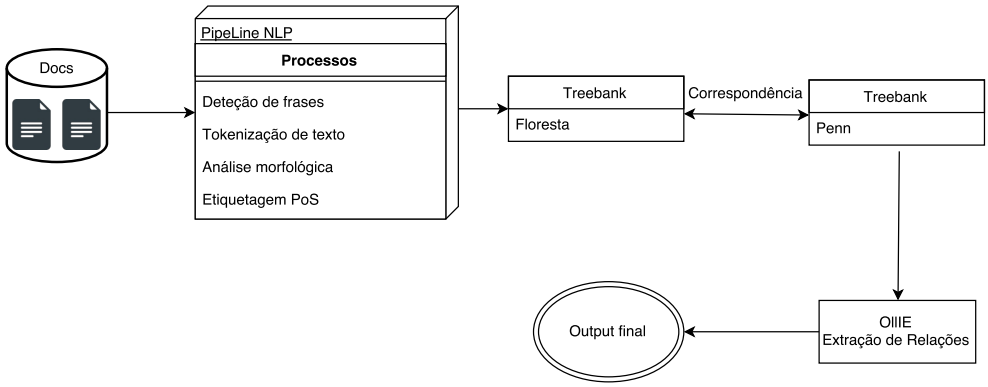

Figura 3.1: Diagrama da inicial

3.3 Seleção de ferramentas para pipeline

Após o estado da arte e aproximação ao objetivo e à extração de informação sob relação aberta, foi importante uma triagem das ferramentas encontradas e testadas e por consequência centrar o trabalho essencialmente nestas de forma a que se optasse pelo melhor caminho. Destaca-se que todas estas ferramentas foram devidamente medidas de acordo com o impacto que teriam no resultado final deste projeto, assim como a compatibilidade entre as mesmas. Vamos então começar por explicar as aplicações escolhidas e o impacto que têm na nossa extração de informação. Numa fase inicial começamos por ter a entrada de texto com informação coerente e que faça sentido, este texto deverá ser na língua portuguesa e ter o formato apropriado, UTF-8 nesta fase, para que caracteres sejam vistos como tal e tratados dessa forma. O processo de extração de informação do conteúdo desejado será colocado em pipeline no nosso software para ser tratado conforme apresentado nos diagramas que representam as soluções previstas.

Para que seja possível o reconhecimento de texto em Português no nosso software é necessário passar por algumas fases, essas são fases de analise morfológica, algumas soluções existem e foram testadas, de seguida se apresentam as duas mais relevantes ao nosso trabalho, será apresentada a sua comparação.

3.3.1 Seleção de anotador morfossintático

A presente comparação centra-se no analisador morfossintático, sendo o principal objetivo a precisão de anotação dos dois sistemas, o FreeLing e o TreeTagger, mais concretamente nas versões 4.0 e 3.2 respetivamente. Estes foram treinados e avaliados com os mesmos recursos linguísticos para uma mais precisa comparação, foi um teste desenvolvido pela Universidade de Santiago de Compostela, Centro Singular de Investigação em Tecnologias da Informação (CITIUS)[18] que nos ajudou na escolha do anotador morfossintático.

Candidatos

O FreeLing[31], possui desempenhos em PoS Tagging próximos do estado da arte existentes, variando sempre entre os 94% e 98% de precisão. Este teste tem como base uma corpora de treino [20] (Garcia e Gamallo, 2010). O módulo usado para proceder a esta análise morfológica foi com base em trigramas[12].

A segunda ferramenta é o TreeTagger[34], cujo método de PoS Tagging se baseia em árvores de decisão e o seu desempenho em experiências anteriores atinge os 96% de precisão para o Inglês e Alemão, contudo para o Português existe um splitter que separa as conotações dos pronomes, além de reconhecedor de entidades mencionadas.

Treino

Os recursos para treinos destas duas ferramentas basearam-se na adaptação do Bosque[7] criado a partir do CETEMPúblico e que contém mais de 138.000 tokens revistos manualmente por linguistas. Este corpus é compilado com base em artigos do jornal Público.

Além do corpus, também se utilizou um dicionário, que contém mais de 1,257.000 de formas, cujo foi extraído a partir do léxico LABEL-LEX(SW)[14].

Tanto o corpus como o léxico foram alvo de algum trabalho de adaptação de etiquetas de um tagset da fonte original para um tagset comum[20]. O tagset usado tem 255 etiquetas e baseia-se nas recomendações do grupo EAGLES[22], conforme explicado anteriormente, é o tagset utilizado para caracterizar o Português.

Método de Avaliação

Como já fora referido, para se conhecer o desempenho de cada sistema de Part of Speech tagging, foram treinados e realizadas avaliações com dois corpus de teste diferentes, foi medida a precisão dos sistemas, que se calcula dividindo o número de tokens etiquetados corretamente pelo etiquetador entre o número total de tokens do corpus.

$$\text{Precisão} = \frac{\text{\# Tokens etiquetados correctamente}}{\text{\# Total de Tokens}}$$

Figura 3.2: Cálculo da precisão do FreeLing e TreeTagger

O corpus empregado para realizar a avaliação dos sistemas foram o Bosque CF, cujo é uma adaptação do Bosque CF 8.0, criado a partir do CETENFolha, que contém 81.000 tokens revisados manualmente por linguistas e cujo foi compilado de artigos do jornal a Folha de São Paulo.

Também foi utilizado o corpus de teste Miscelâneo[18] que é um pequeno corpus de 600 tokens anotado manualmente e especificamente para esta avaliação. Foi compilado a partir de textos da Wikipédia e de excertos de obras literárias Portuguesas.

Para a avaliação foram procurados os tokens que são avaliáveis, estes são tokens que aparecem mais que uma vez e que não podem ser etiquetados com tags diferentes, ou seja, não pode ser ambíguo, dessa maneira não se contabilizam problemas de spliting ou reconhecimento de entidades.

Resultados da Avaliação

Podemos então ver na tabela 3.1 os resultados das avaliações realizadas com o corpus Bosque CF[18]:

Tabela 3.1: Precisão entre FreeLing e TreeTagger em corpus Bosque CF

Sistema	Número de Tokens avaliáveis	Precisão
FreeLing	66.612	93,03%
Treetager	69.118	91,39%

Da tabela acima se pode retirar a precisão entre os dois sistemas de acordo com o número de tokens avaliáveis, se obteve melhor precisão no FreeLing.

Os valores de precisão podem ainda ser melhorados com o novo corpus Miscelâneo, como se apresenta na seguinte tabela 3.2. Este corpus conforme explicado foi criado precisamente pelo CITIUS para este comparativo[18].

Tabela 3.2: Precisão entre FreeLing e TreeTagger em corpus Miscelâneo

Sistema	Número de Tokens avaliáveis	Precisão
FreeLing	532	98,30%
Treetager	529	96,03%

Conclui-se então que o desempenho de dois sistemas de anotação morfossintáticos para o Português, com os mesmos recursos linguísticos, corpora e dicionários que foram utilizados tanto no processo de treino como no processo de teste, os resultados indicam que o módulo de Part of Speech Tagging do FreeLing atinge um melhor valor com diferenças de aproximadamente 2%.

O FreeLing[31] é então usado na sua versão 4.0, contém módulos de analisadores e conteúdos programáticos que foram devidamente alterados para fazer face às tarefas necessárias e que podem ser utilizados para a parte inicial do trabalho, este é o nosso ponto

de entrada de informação e que funciona segundo o esquema presente no Pipeline FreeLing inicial 3.3

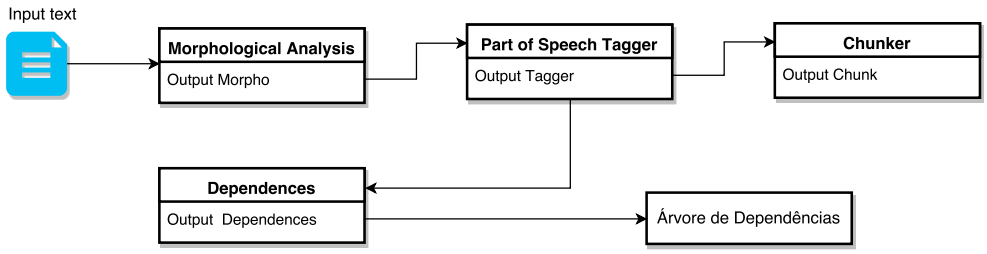

Figura 3.3: Pipeline inicial tendo como base o FreeLing

A execução deste pipeline de ferramentas, que representa um teste inicial do sistema, permite que à saída exista um ficheiro que é tratado no formato ConLL com a devida árvore de dependências e a devida etiqueta que representa cada palavra formando uma árvore de compreensão, o que é importante para se relacionar com o software MaltParser[29].

Sendo assim tomou-se a decisão para a escolha do FreeLing nesta mesma versão, isto numa fase inicial, pois além de um bom PoS Tagger possui outros módulos de interesse que trabalham de acordo com o necessário para o trabalho que se pretende, como se indica explicado no diagrama que se segue 3.4.

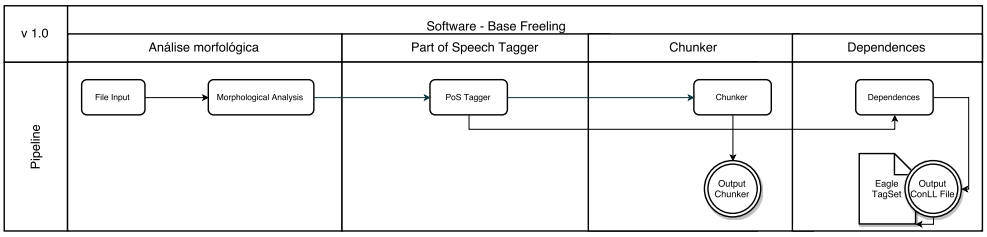

Figura 3.4: Pipeline utilizado numa versão inicial do sistema

Este diagrama exemplifica de uma maneira objetiva a primeira aproximação ao trabalho, o objetivo aqui seria simples, a utilização do output das dependências que tem como base o ficheiro em formato ConLL e que seria dado como argumento de entrada no software MaltParser na sua versão 1.7, com algoritmo linear, para que fosse possível obter um ficheiro de extensão .mco que é dado como entrada no software OllIE, para extração de informação com base no Português.

O OllIE[35] é usado neste trabalho como base com as suas classes e subclasses inerentes a todo o processo de extração. Este é um software para programadores que automaticamente identifica e extrai relações binárias de frases em Inglês, daí ser necessário todo este processo com o FreeLing, para uma tentativa de extração para português, com auxilio do ficheiro que se consegue com o MaltParser. O OllIE foi projetado para a extração de informações

onde as relações de destino não são especificadas com antecedência, portanto um sistema de extração de foco aberto.

Este sistema está licenciado sob a General Public License[23] (v2 ou posterior), cuja a fonte está incluída assim como os componentes para a chamada em linha de comando de uma Java API. Licenciamento de código aberto está sob o licenciamento completo GPL, que permite muitos usos livres, dos quais se insere o uso para trabalhos e projetos académicos como esta dissertação.

Aqui, e depois dos passos anteriores terem sido devidamente ultrapassados, estamos prontos para o processo de extração, vamos então usar o ficheiro de dependências em formato ConLL, com as devidas alterações para que possa ser consumido pelo MaltParser e posteriormente consumido pelo OllIE e analisar se existe extração.

Contudo tal não se provou possível como seria de esperar, apesar de termos um ficheiro pronto para dar entrada no maltparser, no seu formato ConLL, este possuía algumas imperfeições e campos que não eram lidos devidamente pelo MaltParser, nomeadamente campos nulos, além de que o corpus utilizado para a extração desse ficheiro era muito pequeno e portanto pouco eficaz, também era sabido à partida que o OllIE, funcionava apenas com o tagset Penn Treebank e o tagset utilizado no FreeLing era o EAGLES. Sendo assim, teríamos de encontrar uma nova maneira e método para gerar um modelo de parsing aceitável, com um corpus de entrada no MaltParser suficientemente grande para produzir uma extração de informação viável e programar uma aplicação que lidasse com esse aspeto, esta que teria de utilizar a API do OllIE e ser feito à medida das nossas necessidades.

Aqui deparamo-nos com um problema, ou seja, arranjar um ficheiro suficientemente grande que pudesse ser já em formato ConLL para dar entrada no MaltParser, e com isso poupar tempo de processamento com o FreeLing para obter um, que teria de ser obtido através de um corpus suficientemente grande. Como enumerado na secção anterior, a Floresta Sintática possui um ficheiro ConLL com o tagset EAGLES em que cujo o texto que o produziu fora devidamente anotado por linguistas e que advém do CETEMPúblico, é portanto um corpus enorme e que será a solução mais viável.

Sendo assim, é criado um ficheiro de extensão .mco com esse modelo induzido com o treebank EAGLES associado, usado para que se consiga proceder à devida analise do texto Português e como tal ter uma base com alguma credibilidade.

Já se havia falado no ficheiro de formato ConLL, mas ainda não se explicou bem o seu significado no contexto de EI.

Um ficheiro em formato ConLL é um modelo pré-treinado com um sistema de parsing de dependências de dados, que é usado para induzir um modelo de análise de em formato Treebank. Pode ser tirado à saída do pipeline FreeLing, dado um corpus de entrada ou pode ser um ficheiro cujo texto já fora anotado por linguistas. O formato ConLL tem a seguinte disposição presente na tabela 3.3.

Após esta decisão, e tendo já o ficheiro .mco de saída do MaltParser anotado com o tagset EAGLES, precisávamos de uma melhor análise e processamento de texto que ligasse de alguma forma com o extrator de informação utilizado, poupando tempo de processamento e análise, foi então escolhido o OpenNLP[30], este possui um conjunto de

Tabela 3.3: Formato ConLL

ID	Index na frase, a começar em 1
FORM	Palavra por si só
LEMMA	Lema ou haste da palavra
PLEMMA	
POS	Part of Speech
PPOS	
FEAT	Lista de características morfológicas se paradas por |
PFEAT	
HEAD	Index do parente sintático, sendo 0 a raiz
PHEAD	
DEPREL	Relação sintática entre a HEAD e a palavra
PDEPREL	

ferramentas usadas para derivar informações úteis e significativas de fontes de linguagem natural, especialmente documentos de texto, como é o nosso caso.

O OpenNLP vem da Apache Foundation e é uma biblioteca JAVA, o que nos interessa pois forma uma API que pode ser incluída na nossa ferramenta sem ter se ser partida em módulos de entrada para pipeline e cuja é de código aberto, fornecendo serviços como Sentence Detection e Segmentation, PoS Tagging, Named Entity Extraction, Chunking, Parsing, que já foram explicadas, porém para a língua Inglesa. Dado este problema, foi necessária e utilização de novos modelos para processar a língua Portuguesa, obtidos nos repositórios do OpenNLP. De seguida se indicam os processos implementados.

3.4 Implementação

O OpenNLP, como já fora referido, tem características que nos foram muito úteis, aqui se explicam as utilizadas no nosso software, assim com partes de código que permitem essa funcionalidade:

- Sentence Detector é utilizado para detetar os limites e margens de uma dada frase, dado um determinado paragrafo. É usado o ficheiro binário "pt-sent" que permite ao Sentence Detector um ficheiro já treinado em ConLL pelo conteúdo Bosquee.

```
InputStream modelSent = new FileInputStream(``data/pt-sent.bin'');

     modelS = new SentenceModel(modelSent);
     if (modelSent != null)
         modelSent.close();

    sentenceDetector = new SentenceDetectorME(modelS);

public String[] detect(String arg){
   return sentenceDetector.sentDetect(arg);
```

Inicialmente é criado um inputStream com o ficheiro para deteção de frases em português, retirado das fontes do OpenNLP, depois com base nesse modelo é dada a entrada do texto a ser analisado, o retorno é dado por uma frase por linha do texto que entra.

- Tokenizer, que separa o que foi dado como frase em palavras por linha, tendo como restrição a identificação a partir de espaços que separam cada palavra. Iremos ter portanto uma palavra por linha, incluindo pontuação e usando o binário pt-token, treinado com ConLLx talbanken05.

```
InputStream modelToken = new FileInputStream(``data/pt-token.bin'');
     modelT = new TokenizerModel(modelToken);
     if (modelToken != null)
         modelToken.close();

    tokenizer = new TokenizerME(modelT);

public String[] tokenize(String sentence)
    return tokenizer.tokenize(sentence);
public TokenizerModel getModel()
    return modelT;
public opennlp.tools.tokenize.Tokenizer getTok()
    return tokenizer;
```

O tokenizer funciona de forma muito parecida ao anteriormente explicado, no entanto aqui é retornado ao programa principal uma lista das palavras que formam as frases anteriormente divididas.

- Part of Speech Tagger, que é usado para dividir o texto nos vários elementos gramaticais existentes no mesmo, para uma análise mais aprofundada. Usando o binário para Português pos-maxent, modelo treinado com ConLLx Bosque .

```
InputStream modelTag = new
    FileInputStream(``data/''+lang+``-pos-maxent.bin'');
      modelTa = new POSModel(modelTag);
      if (modelTag != null)
          modelTag.close();

    tagger = new POSTaggerME(modelTa);

public String[] tag(String[] text)
    return tagger.tag(text);
public POSModel getModel()
    return modelTa;
public POSTaggerME getTagger()
    return tagger;
```

Aqui podemos ver o segmento de part of speech, cada token extraído anteriormente é etiquetado de acordo com o tagset EAGLES.

Como já fora referido, o OllIE foi a ferramenta escolhida para EI, dado que nesta fase já se teria o texto devidamente anotado e pronto a ser analisado utilizando o OpenNLP. O OllIE está formatado para aceitar somente tagset de Penn TreeBank[25] o que faz com que se tenha de manipular e mapear o tagset EAGLES com o tagset exposto para se conseguir o objetivo de extração em Português, tendo sido utilizada a seguinte aproximação.

Foi usada uma solução que tem por base a utilização de um programa desenvolvido em JAVA que utiliza o pacote do OpenNLP e que também inclui o pacote de parsing Português treinado com MaltParser[29] com base num documento apresentado em formato ConLL da Floresta Sintática[7].

Aqui a engenharia aplicada concentrou-se num mapeamento o mais direto e objetivo possível do Tagset da Floresta Sintática para o Tagset Penn Treebank e que ocorre em runtime de uma qualquer execução de extração e que resultará numa análise do OllIE ao documento do qual se pretende extrair informação, de onde será depois retirado o grafo associado. Estão então reunidas as condições para que haja uma extração de informação aceitável na língua Portuguesa.

Esta aproximação revelou-se a mais precisa e fiável para se melhorar, pois além de se relacionar bem com o output do MaltParser, apresenta modelos para open parsing, ou seja, é possível modificar e/ou adicionar novas relações de parsing e com isso trazer a possibilidade de mais e melhor extração de informação, com base nas novas relações. Este processo será explicado na próxima secção, sendo também dado um exemplo do tipo de relações adicionadas. Também o seguinte diagrama mostra o aspeto final no que toca à seleção de ferramentas.

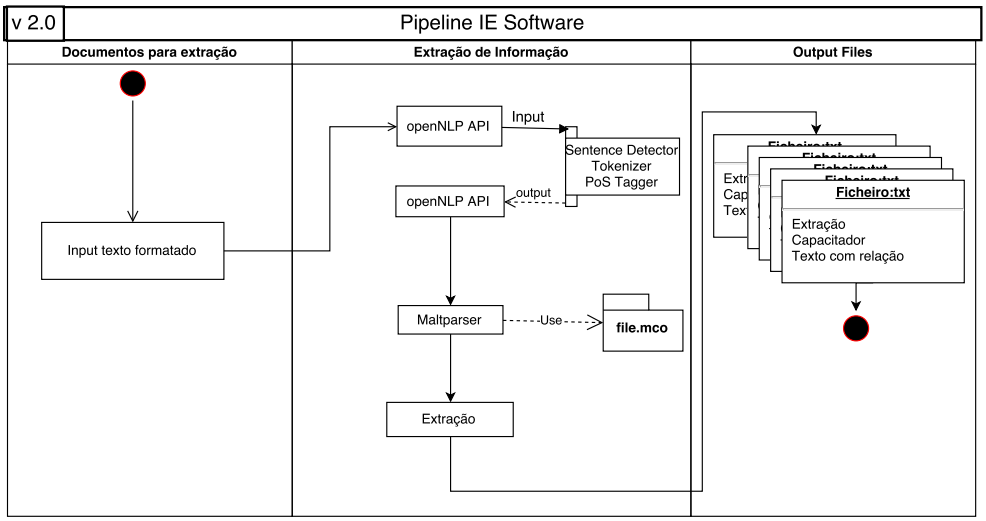

Figura 3.5: Diagrama da Framework geral do software numa versão 1.0 - primeira versão com o OllIE

3.4.1 Módulos do Pipeline

Para dar uma introdução sobre os módulos utilizados vamos ter como auxílio diagramas que nos explicam como funcionam internamente, assim como parte de algum código que os faz funcionar dessa forma. Estes módulos são, como especificado, os módulos necessários para que haja uma extração de informação viável apenas, sendo que nas seguintes experiências foram aprimorados e alterados de acordo com o que seria suposto. Será também tida em conta os detalhes de implementação de cada módulo, sendo estes explicados em cada sub secção. Foi dividido em três módulos para mais fácil explicação, também porque no próprio software existe uma mais fácil perceção entre módulos se dividirmos o programa nestas três partes, outra das razões desta divisão é também porque os outputs de cada modulo já foram explicados.

3.4.2 Módulo 1 - Separação e anotação morfossintática das palavras

Módulo inicial do pipeline da versão 1.0 do software, esta verão já conta com o auxilio do OllIE , MaltParser e ficheiro .mco de acordo com as regras para português. Se segue explicada de seguida.

Este módulo trata dos dados à entrada do nosso software de EI, os quais não têm o problema de versões anteriores que requeriam somente ficheiros tratados como UTF-8, aqui o nosso software faz a verificação e traduz para o caso de não ser o formato desejado, UTF-8.

```
byte[] ptext = String.getBytes(ISO_8859_1);
```

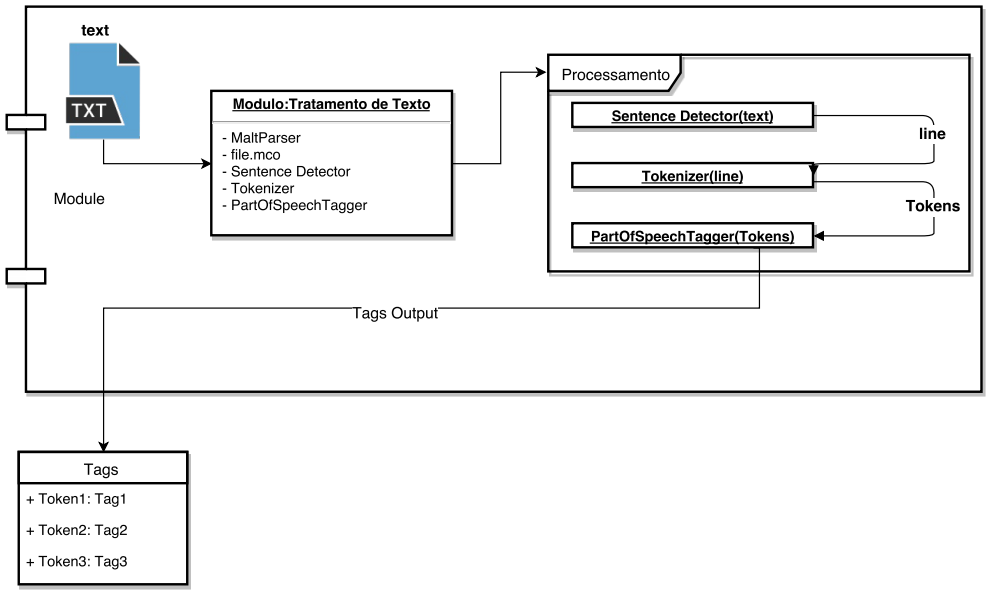

Figura 3.6: Módulo 1 - Separação e anotação morfossintática das palavras

---

```
String textval = new String(ptext, UTF_8);
```

Estes dados são tratados agora pelo OpenNLP, onde é feito o detetor de frases e posteriormente a tokenização das mesmas, sendo essas classificadas través de etiquetas no formato Floresta Sintática, este tipo de etiquetas é usado para anotar o texto português, de modo que foi necessário que o MaltParser possuísse o ficheiro ConLL devidamente tratado e anotado para que o nosso texto não sofresse alterações e fosse consequentemente bem anotado.

```
SentenceDetector sd = new SentenceDetector();
Tokenizer tokeniz = new Tokenizer();
POSTagger postagger = new POSTagger(‘‘pt’’);
```

É portanto criada a lista de palavras ou tokens com a devida anotação, como mostra o seguinte exemplo, para darem entrada no próximo modulo 3.7.

```
Token:Fatores - tag:n
Token:Demográficos - tag:prop
Token:e - tags:conj-c
Token:Económicos - tags:prop
Token:Subjacentes - tags:adv
```

3.4.3 Módulo 2 - Tratamento dos tokens e tags por parte do MaltParser para atribuição de etiquetas

Este módulo, recebe os tokens já previamente anotados, pois após executar o tokenizer, o PoS tagger tratou de etiquetar cada um corretamente. De seguida entramos numa preparação para extração, onde é inicializado o MaltParser.

```
MaltParserService mps = new MaltParserService();
String tokens[] = tokeniz.tokenize(line);
String tags[] = postagger.tag(tokens);
String []tok = new String[tokens.length];
```

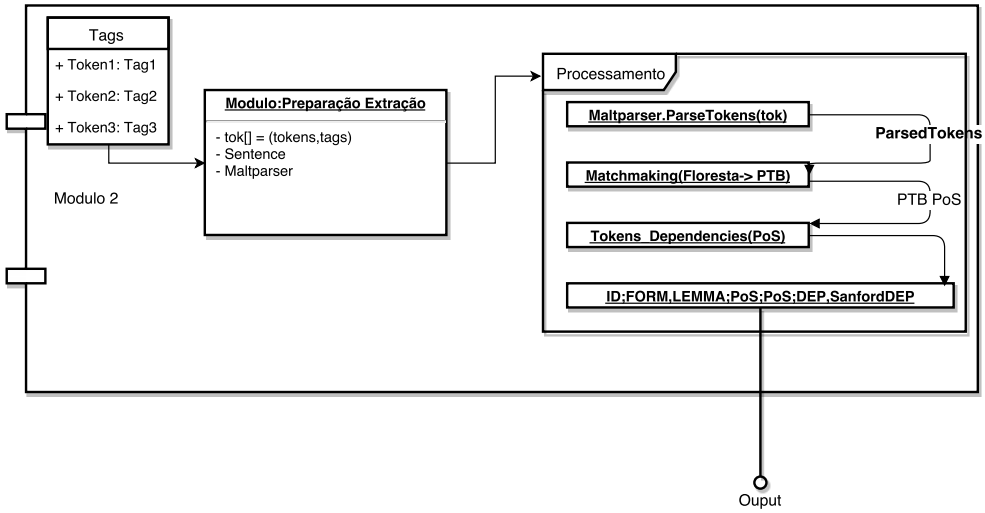

Figura 3.7: Modulo 2 - Tratamento dos tokens e tags por parte do MaltParser para atribuição de etiquetas

É depois feito o parsing de cada token e feita a atribuição do respetivo tag Penn Treebank, como mostra o exemplo e com isto que se forme o ficheiro já contendo as dependências para a terceira fase do nosso software, que o OllIE requer para que haja extração.

```
1  Fatores Fatores NN NN 0 _
2  Demográficos  Demográficos  NNP   NNP   1  prep
3  e  e  CC CC 2 cc
4  Económicos Económicos NNP  NNP   2  null
5  Subjacentes Subjacentes RB RB 9 advcl
```

3.4.4 Módulo 3 - Entrada de etiquetas e dependências para extração de informação com base em modelos

Finalmente chegamos ao modulo 3, este que recebe o ficheiro associado ao texto a ser analisado devidamente anotado com atribuição de etiquetas para o formato Penn Treebank e com as dependências, na qual a "string" representará cada token desse mesmo texto, de seguida é feita todo o processo de padrões de decisão ao estilo do ReVeb pelo OllIE, cujo aplica os modelos de extração, que serão abordados numa versão posterior do software.

```
HashTable ht=new HashTable();
ht.put(‘‘n’’, ‘‘NN’’);
ht.put(‘‘n-adj’’, ‘‘JJR’’);
ht.put(‘‘adj’’, ‘‘JJ’’);
(...)

HashTable dt=new HashTable();
dt.put(‘‘SUBJ’’, ‘‘nsubj’’);
dt.put(‘‘ACC’’, ‘‘dobj’’);
dt.put(‘‘ACC-PASS’’, ‘‘nsubjpass’’);
String[] tt = ds[i].split(‘‘\\t’’);
(....)

string += tt[0]+‘‘\t’’+tt[1]+‘‘\t’’+tt[2]+‘‘\t’’+ht.get(tt[3])+
\t’’+ht.get(tt[4])+‘‘\t’’+tt[5]+‘‘\t’’+dt.get(tt[6])+‘‘\n’’;
```

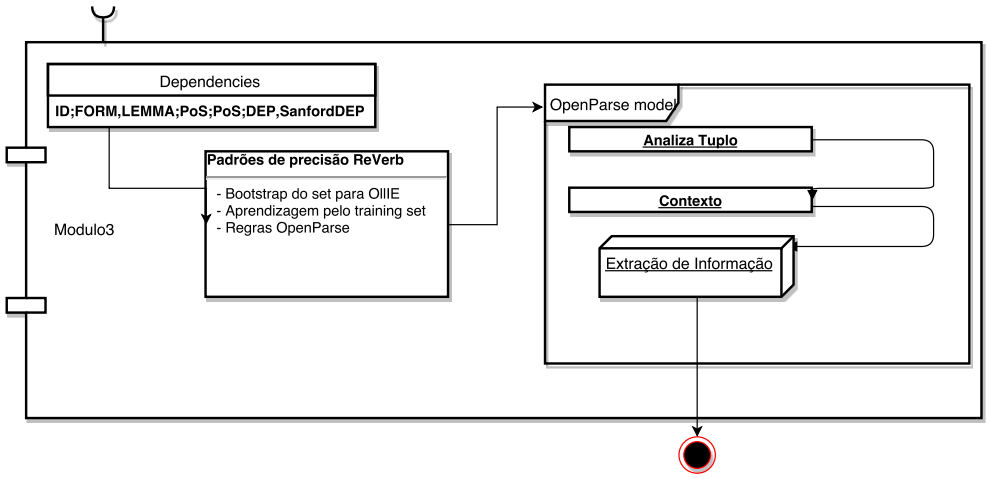

Figura 3.8: Modulo 3 - Entrada de etiquetas e dependências para extração de informação com base em modelos

Aqui já temos um output que contém uma extração, de acordo com o texto e regras presentes no ficheiro de regras. Estas regras são no fundo um ficheiro de modelos que

são extremamente importantes para que haja efetivamente extração relacionando os tuplos entre si de acordo com o especificado nesse mesmo modelo. Estes modelos estavam em inglês e eram extremamente extensos, portanto, após uma limpeza ao ficheiro e analisadas as regras que seriam de aproveitar para se obter alguma objetividade na extração, foram traduzidas as aproveitadas e criadas novas, que complementassem a nossa extração.

As extrações são retiradas a partir do OllIE com base nesse ficheiro de modelos, que com as regras induzidas pode iterar o texto e captar o tuplo, o qual produz uma informação extraída. De seguida se apresenta um exemplo do código que trabalha no sentido da extração.

```
// formado grafo de dependencias
DependencyGraph graph = DependencyGraph.fromCONLL(com);
//criado o objecto para OllIE API
Ollie ollie1 = new Ollie();
//OllIE vai analisar o grafo e extrai-lo de acordo com as regras
ollie1.extract(graph)=graph;
//itera cada instancia do grafo para produzir uma analise coerente
Iterable<OllieExtractionInstance> extrs =
scala.collection.JavaConversions.asJavaIterable(graph);
for (OllieExtractionInstance inst : extrs) {
OllieExtraction extr = inst.extr();
//por fim, de cada inastancia se extrai os argumentos e relacao
 extr.openparseConfidence() + ‘‘\t-’’+ extr.arg1().text()
 +‘‘\t-’’+extr.rel().text()+‘‘\t-’’+extr.arg2().text());
 }
```

Podemos ver presente na extração “extr” o argumento1, relação, seguido de argumento 2, esta extração é levada a cabo por um conjunto de modelos presentes no ficheiro openparse que contem os modelos para extração de relações, ou seja, as regras referidas aqui. Esse modelo foi crucial para as novas versões do software, como a seguir se apresenta.

Capítulo 4

Aplicações, Testes e Experiências

Tendo como base assente um teste inicial da qual fazia parte o pipeline com os módulos devidamente alterados do FreeLing em que no seu cômputo geral teria como saída um ficheiro que seria validado e dado como entrada no software MaltParser, este denotado de teste ou versão preliminar.

O MaltParser contém opções de algoritmia pré definidas para fabrico de um modelo de parsing, .mco, necessário para o reconhecimento da língua Portuguesa, concluímos a não utilização devido não só à estrutura do ficheiro .ConLL utilizado, que continha espaços nulos e alguns campos que não eram relevantes, mas também pelo tamanho do corpus que seria necessário para criar um ficheiro suficientemente grande, foi descartada esta opção e avançada a versão 1.0.

Por forma a avaliar as capacidades, limitações e alcance do sistema de extração proposto, serve a presente secção para dar a conhecer as várias versões do software. Para isso foram definidas duas versões, a versão 1.0 que foi sendo melhorada e a versão 2.0 que testa experiências finais com recurso a documentos da Coleção Dourada do Segundo HAREM

Sendo assim a primeira experiência, que podemos denotar de versão 1.0, teve como objetivo testar o funcionamento do software programado que trabalha com a API OllIE, OpenNLP e utilizado o ficheiro ConLL da floresta sintática, ao invés do ConLL de saída do freeling, usado no MaltParser. Na experiência 1 e 2, ou versão 1.0 busca uma aproximação mais focada e programável de EI, esta tem diferentes modificações no que diz respeito tanto a módulos de processamentos como em saída de informação. De seguida se apresenta a versão 2.0 que prima por operar com o XML da CD HAREM com foco, resultados e respetivas conclusões.

Estes testes são de extrema relevância, pois caso não seja explicito uma saída mínima e aceitável, teriam de ser alteradas as metodologias de extração ou mesmo o core da aplicação. Caso fosse produzido uma saída mínima aceitável de informação que nos explicasse, numa base limite de dois argumentos e uma relação que as liga, estaríamos na correta direção para proceder a ajustes do software e com isso a realização de novos testes e comparação dos mesmos de modo a contabilizar um avanço na matéria.

4.1 Corpus para testes

Para proceder a testes, foi necessário um corpus devidamente tratado, foi escolhido um conjunto de notícias extraídas de jornais online, como o Diário de Notícias, Jornal de Notícias e Observador, que vão ser dados como entrada e que serão alvo de uma análise progressiva que resulte em testes propriamente ditos de modo a contabilizar o software em termos de performance e resultado, são depois guardados no diretório raiz do software e devidamente enumerados e etiquetados, para que se distingam entre si e façam a ligação direta ao texto que originou essa extração.

Para levar a cabo estes testes teremos de reunir a informação que queremos analisar, vamos ter dois tipos de aplicação, ambos são de extrema importância para analisar resultados e perceber as melhores aproximações, num primeiro caso optamos por notícias do âmbito de empresas e economia, podendo ser este qualquer outro tipo de informação. Estes textos foram reunidos com recurso a uma ferramenta de extração de links que se denota de import.io, a qual já contém um crawler que reúne todos os links referentes às notícias e que os extrai para uma tabela, de seguida é feita manualmente a cópia dessas notícias para ficheiros de texto, para serem tratados posteriormente.

Ainda neste âmbito foram reunidos um total de vinte e um textos para a primeira fase de testes, sendo que para esta análise foram utilizadas apenas as primeiras sete notícias, extraídas do Jornal de Notícias, que vão ser dadas como entrada ao software aqui testado, este dará como saída um conjuntos de ficheiros de acordo com o número de ficheiros de entrada, neste caso sete, com a informação extraída de cada um. Com este teste espera-se a possibilidade de reunir factos na extração que resumam a notícia mesmo que de forma pouco granular.

4.2 Sistema 1

Nesta versão temos um software que prima por ter a particularidade de ser programável em cada módulo o que faz dele uma aplicação escalável e configurável. Este contém os módulos necessários para tratar a informação recebida, que já foram explicados no decorrer desta dissertação, são de extrema importância para que tudo funcione bem, como o Sentence Detector, Tokenize, PoS Tagger desenvolvidos a partir da API OpenNLP e ficheiro .mco usável que tinha sido possível dada a entrada do ficheiro ConLL disponibilizado pela Floresta Sintática, este ficheiro .mco era então extraído a partir do MaltParser, o que possibilita o reconhecimento do Português.

Esta versão conforme se apresenta, contém o ficheiro de modelos de extração para Português, de modo a que as palavras sejam reconhecidas e inseridas num dos modelos de extração, sendo que é contabilizado com o valor da confiança de cada possível relação, isto quer dizer que se existir uma confiança relativamente alta de determinada extração após analisado um parágrafo, essa relação é aceite como válida e expressa em resultado de saída. De frisar que todo o processo de tradução de modelo foi levado a cabo para se proceder a este teste inicial.

Dado como entrada o corpus que é composto pelos ficheiros de notícias acima discriminados e vamos analisar alguns deles e posteriormente obter resultados, para demonstrar se existiu sucesso de extração ou não.

4.2.1 Experiência 1 - Extração com a versão 1.0 do sistema

Esta primeira experiência tem como objetivo proceder a uma extração de informação de texto apresentado, para isso foi levada a cabo uma análise textual com a API OpenNLP, fazendo uso dos seus módulos, com o ficheiro extraído pelo MaltParser cujo forma o modelo de parsing .mco e que tem como base o ficheiro .ConLL da floresta sintática. De seguida é feita a relação entre o treebank usado pela floresta sintática, EAGLE, e o treebank usado pelo OllIE, Penn Treebank, obtendo por fim uma extração de acordo com as normas e regras presentes no ficheiro modelo de extrações e posteriores ficheiros de ligação de tuplos do OllIE, devidamente traduzido para o Português

Espera-se então como objetivo desta primeira análise quantitativa com o sistema e adaptação das regras do OllIE, uma extração coerente que formule uma visão geral do que é dado como entrada do extrator de informação aqui presente, sendo posteriormente analisados os triplos de saída e observando a relação entre eles.

Estes testes mostram a motivação necessária para se continuar o trabalho e sobretudo se estaríamos no caminho correto ou não para extração de informação em Português de acordo com os resultados obtidos ou se teríamos de envergar por outra solução. Ajustes e alterações de metodologias poderiam ter de ser tomadas em conta para que obtivéssemos um melhor comportamento, contudo nesta fase seria apenas relevante o comportamento do software e se estávamos a ter relações válidas entre dois argumentos, essa quantificação é feita da seguinte forma: Sem saber o que estava explicito no argumento de entrada, ou seja, numa determinada notícia que é dada como entrada no software, teríamos de ser capazes de perceber algum assunto que estivesse exposto nessa mesma entrada de informação, ou pelo menos que se soubesse de que se tratava essa informação.

4.2.1.1 Resultados

Para expressar resultados obtidos é adotada a seguinte formula, serão mostrados excertos das notícias analisadas e seus resultados, passando por uma breve explicação do que se trata e analisar a existência de algum tipo de relação entre os dois ficheiros, tanto os de entrada como os de saída com a informação extraída.

Dado inicialmente uma notícia que exprime o seguinte:

(1)*Determinada empresa vai apresentar um documento, após reunião. Esse documento tem a particularidade de conter interesses para a empresa. Dados esses acontecimentos, sabe-se que existe uma greve a decorrer devido ao incumprimento do mesmo.*

Após a passagem no software de extração de informação temos o seguinte resultado:

```
-Este documento -congregará Um documento que representará o sentimento dos
nossos trabalhadores -os interesses\\

-Este documento -congregará porque vai ser o produto de um trabalho que
vamos desenvolver nos próximos dias internamente e também com a
colaboração de diversos tripulantes de cabine -os interesses\\

-A administração -companhia -aérea açoriana\\

-uma paralisação -em -termos
```

Com este resultado, estamos somente em condições de analisar a relação existente entre o texto dado como entrada, neste caso uma notícia, e a extração de informação referente a essa mesma notícia. Podemos ler da seguinte extração que nada nos responde a uma pergunta objetiva relativamente à notícia, sendo portanto uma informação pouco útil e pouco relevante.

Ainda assim podemos afirmar que conseguimos especificar apenas de que se trata essa notícia, mas não o que desencadeia as ações, ou a entidade que a rege assim como o conteúdo da sua informação e algumas características da mesma, contudo é possível extrair algo, ainda que pouco granular. Este é um resultado a ser melhorado num próximo passo.

Prosseguimos para a análise de uma outra notícia de modo a encontrar um padrão de extração.

(2) *A seguinte notícia mostra que uma empresa espanhola com instalações em Matosinhos entrou em greve, toda a restante notícia está em torno do que desencadeou esse acontecimento, neste caso foi a redução salarial, bem como horas extra de trabalho.*

Tendo como extração de informação, após passagem pelo software o seguinte resultado:

```
-Lusa -explicou -fonte
-o pré-aviso -tem -data
-a empresa -decidiu arbitrariamente -diárias
-O tribunal -deu -razão
-Fernando -recorreu -a motoristas
```

Esta extração não é minimamente útil, não se pode afirmar nenhuma informação reveladora de greve, nem o que despoletou a mesma. A única informação útil que podemos tirar é que a lusa veio explicar algo à fonte, que existe um pré aviso e uma data, que o tribunal deu razão a algo e que uma entidade recorreu a motoristas. Não temos nenhuma informação de greve, nem nenhuma razão para essa acontecer, o que mostra uma possível falha no sistema, contudo essa falha pode ser eliminada com uma reformulação dos modelos de extração tal pode ser visto no segundo teste executado ao sistema, onde se prevê que os resultado sejam melhores.

Os resultados apresentados mostram que a adaptação ao Português não é uma tarefa trivial.

4.2.2 Experiência 2 - Extração de informação de notícias de jornais com Versão 1.0 do sistema

Com esta experiência o que se pretende é melhorar a versão 1.0 com vista a eliminar alguns problemas existentes, como é o exemplo um (1) dado na secção anterior em que não é clara a informação extraída, não se percebe ou compreende objetivamente de que se trata ou a entidade que a rege e a informação que é pouco granular.

Ou o exemplo dois (2) da mesma versão, em que o resultado da extração não mostra objetivamente de que se trata a informação inicial, nem o que desencadeia essa mesma informação na sua raiz.

Para colmatar esses defeitos foi necessário proceder a algumas alterações programáticas no que toca ao ficheiro openparse que contém os modelos de extração de informação e internamente no próprio software no que diz respeito tanto ao formato de saída como à relação entre treebanks, para que com essas novas definições fosse possível o levantamento de novas extrações.

Vamos analisar o ficheiro de modelos openparse.model, que como já fora explicado é o ficheiro que contém as relações possíveis de serem tidas como informação entre dois argumentos. Este ficheiro além de ter sido alvo de uma tradução intensa no que toca aos modelos existentes, foi alterado de modo a que fossem adicionadas mais relações para atingir mais e melhores extrações e com isso colmatar alguns problemas existentes.

O seguinte exemplo de algumas relações adicionadas a ficheiro e a extração que produziram é apresentado, para ser mais fácil a identificação de resultados positivos, vamos analisar os mesmos textos informativos usados anteriormente para que seja possível a comparação:

```
Opeparse.model:
R1{rel} arg1} <nsubj< {rel} >dobj> {arg2} 0.1443
R2{rel} {arg1} <nsubj< {rel:postag=VBZ} >prep> {arg2:postag=NNP}0.1443
R3{rel} {arg1} <nsubj< {rel1:postag=VB} >xcomp> {rel2:postag=VB}
>dobj> {arg2} 0.1473
R4{rel}{prep} {arg1} <nsubj< {rel:postag=VBZ} >dobj>{slot0:postag=NN}
>{prep:regex=prep_(.*)}> {arg2} 0.1443
```

Cada uma destas relações (R1,R2,R3 e R4) tem um determinado padrão que olha para cada um dos argumentos dos extremos e os relaciona por termo de uma relação que muitas das vezes é um verbo, contudo essa relação pode ser aberta e aceitar não só verbos como qualquer outro tipo de forma entre dois argumentos. A nossa primeira relação R1 indica que o primeiro argumento tem de ser do tipo nsubj seguido de uma relação à qual está ligado, esta que está em aberto, podendo ser como já explicado de qualquer tipo, em seguida um argumento final do tipo dobj, que também está diretamente ligado à relação. Na segunda relação R2, ainda que um pouco à imagem da relação R1 em que o argumento inicial é um nsubj, já especificamos o tipo de relação que queremos, neste caso é um VBZ que é portanto um verbo finito, seguindo de um argumento final do tipo prep e sendo NNP, ou seja, um nome próprio. Na relação R3 temos duas relações verbo que ligam dois argumentos, em que no ultimo argumento temos de ter presente o tipo dobj. De notar

que não é possível deixar o tipo de argumento em aberto, tendo este de ser previamente especificado. Na última relação usada para exemplo de um ficheiro openparse de modelo de extração temos a relação R4, esta já é um pouco mais complexa pelo que obtém um um argumento do tipo nsubj, uma relação verbo finito, um slot nome do tipo dobj e um argumento final que pode ser qualquer proposição.

4.2.2.1 Resultados

De seguida apresentam-se exemplos de extração obtidos através da notícia que se apresenta, conjuntamente com enabling conditions e texto de onde é extraída. Para facilidade de leitura foram retirados os valores de confiança das extrações, assim como o modelo que as extrai.

> "Motoristas de empresa espanhola com instalações em Matosinhos em greve Os motoristas de uma empresa de transportes espanhola com instalações no Freixieiro, Matosinhos, estão em greve desde segunda-feira para reivindicar melhores condições de trabalho e o pagamento de horas por serviços internacionais, explicou à Lusa fonte sindical. Em causa está a Transportes Martinez e o pré-aviso de greve tem data de segunda-feira vigorando até sexta, dia em que uma comitiva de trabalhadores e responsáveis sindicais vai deslocar-se à Autoridade para as Condições de Trabalho (ACT) para "denunciar a situação". Em declarações à agência Lusa, o coordenador do Sindicato dos Trabalhadores de Transportes Rodoviários e Urbanos do Norte (STRUN), Fernando Oliveira, referiu que "a empresa decidiu arbitrariamente reduzir parte dos salários, bem como o pagamento de duas horas extra diárias por serviços internacionais, conforme prevê o contrato coletivo de trabalho". "Esse valor foi negociado há muitos anos mas, invocando as medidas de austeridade e a crise, várias empresas deixaram de pagar. O tribunal já deu razão a trabalhadores de outras empresas em situação semelhante e agora em causa está esta empresa do Freixieiro", descreveu Fernando Oliveira. A mesma fonte indicou que a greve teve início com cerca de 30 motoristas, mantendo-se hoje com "pelo menos nove", uma vez que, disse o responsável, "alguns cederam ao serem pressionados pela empresa". Fernando Oliveira indicou que "até às 20:00 de ontem [quarta-feira] não saiu nenhum camião [de transporte de gás líquido para abastecer fábricas e hospitais] mas, entretanto, a empresa recorreu a motoristas espanhóis para fazerem o serviço, de forma a que a greve não tenha impacto". O responsável sindical acrescentou ao leque de reivindicações a falta de condições das infraestruturas. A agência Lusa tentou contactar telefonicamente a empresa mas até ao momento não foi possível obter um esclarecimento."

Com base nesta extração que se mostra muito mais completa, a alteração no código para extrair o tipo de desencadeamento que faz com que essa relação seja válida, o que neste output é descrito como "Enabler", teve um grande impacto para processamento da

Tabela 4.1: Exemplo1

Extração #1	Lusa -explicou -fonte
Enabler:	- Some(EnablingCondition(à))
Text:	- Lusa explicou fonte
Extração #2	o pré-aviso -tem -data
Enabler:	- Some(EnablingCondition(vigorando,até sexta , dia,15)))
Text	- o pré-aviso tem data
Extração #3	-a greve -teve -início
Enabler:	- Some(EnablingCondition(com,cerca de 30 motoristas)))
Texto:	a greve teve início
Extração #4	-O tribunal -deu -razão
Enabler	- Some(EnablingCondition(já))
Text	O tribunal deu razão
Extração #5	-a empresa -decidiu diárias -arbitrariamente
Enabler:	- Some(EnablingCondition(bem,como o pagamento de duas horas extra diárias por serviços internacionais , conforme prevê o contrato coletivo de trabalho)))
Text:	a empresa decidiu diárias arbitrariamente
Extração #6	-a empresa -recorreu espanhóis -espanhóis
Enabler:	- Some(EnablingCondition(entretanto,, a empresa recorreu a motoristas espanhóis para fazerem o serviço , de forma a que a greve não tenha impacto)))
Text:	- a empresa recorreu espanhóis espanhóis
Extração #7	-a greve -a -impacto
Enabler:	- Some(EnablingCondition(não,))
Text:	- a greve a impacto

informação, dando mais bases que sustentam a extração. Vamos analisar esta informação sob o ponto de vista de uma pessoa que está somente a ler esta informação e perceber o que se quer lá transmitir.

Analisando os resultados por ordem de extração, que faz sentido pois é coerente com a informação presente no texto inicial: “A Lusa veio explicar à fonte, que existiu um pré aviso com data vigorando até ao dia de sexta feira de uma greve, esta que se fez avançar e cuja teve inicio com 30 motoristas. O tribunal já veio dar razão, supostamente pela greve, pelo que é descrito de seguida que a empresa decidiu pagar as horas extra, como previa o contrato coletivo de trabalho. Contudo sabe-se que a empresa recorreu a motoristas espanhóis para fazerem o serviço, de forma a que a greve não tenha impacto.” Esta ultima extração (6)

não é muito objetiva para o utilizador comum e portanto pode ser mal interpretada, no entanto o "enabler" nega o que lá é dito.

Com esta extração foi possível analisar com granularidade a informação presente numa notícia, o que demonstra algum avanço na extração de informação sobre relação aberta com base num bom e formulado ficheiro de modelos. Além de uma melhor extração, com esta nova aproximação temos o tipo de desencadeamento, descrito como "Enabler", assim como um ficheiro openparser de modelos de extração muito mais aperfeiçoado e como tal muito mais rápido a iterar, pelo que a velocidade de processamento se faz sentir, ainda que se tivesse pouco texto de entrada.

Estas experiências apresentadas serviram para nos forcarmos em aspetos relevantes em cada uma com vista a melhorar até à versão que se apresenta. Por fim foi tentada uma utilização com base na Coleção Dourada HAREM ReRelEM.

4.3 Sistema 2

Tendo nesta fase a versão 1.0 do sistema que conta com extração de informação de notícias como teste, assim como avanços obtidos até chegar à mesma, obtendo e explicando os resultados, passamos à versão de software 2.0.

Ao invés do que se passava nas experiências mostradas, em que o sistema de tokenização e separação das frases selecionava as palavras que deveriam estar juntas e serem vistas como nomes próprios ou entidades relacionadas e as dividia em multi palavras que eram vistas como palavras independentes e que não tinham qualquer relação entre si ou em alguns caso não faziam qualquer sentido, neste sistema esse feito não acontece e as palavras são reconhecidas como entidades e respetivo valor, como se segue no seguinte exemplo:

Nome	Entidade Mencionada	Valor
Reforma Protestante	Reforma_Protestante	ABSTRACCAO
Cativeiro Babilónica da igreja	Cativeiro_Babilónica_da_igreja	ACONTECIMENTO
John Gutenberg	John_Gutenberg	PESSOA

A tabela anterior mostra como exemplo a entidade "Cativeiro Babilónica da igreja", que na versão anterior era vista como quatro palavras diferentes, o que iria estragar a extração, no novo software esse problema é resolvido, sendo que "Cativeiro Babilónica da igreja" é vista como um acontecimento.

Como neste caso vamos ter as entidades mencionadas ou EM do ficheiro XML da Coleção Dourada HAREM no próprio texto a ser enviado para extração, vamos também ter acoplado a esse texto o seu tipo, neste caso vamos ter o NER perfeito em que cada entidade será reconhecida, como já fora mostrado.

O objetivo seguinte será criar no ficheiro que contém os modelos para extração um modelo objetivo que consiga captar e reconhecer cada entidade e o seu valor e assim proceder à extração de informação tendo em conta esse novo valor, que é o seu identificador.

Para isso teremos de proceder a alteração que dará o novo programa, no qual obtemos as entidades mencionadas no texto explicitas com um valor correspondente que a caracteriza, como 'tt0' para a primeira entidade mencionada, 'tt1' para a segunda entidade mencionada e assim sucessivamente, em que o valor irá sendo incrementando à medida que é encontrada cada uma nova entidade ou identificador, este método é necessário para fazer corresponder o novo TAG ao token que é entidade mencionada, como TEMPO, PESSOA, etc.

Esta aproximação deve-se ao facto de ter mais que uma EM com o mesmo nome, mas que possui conotações diferentes ao longo do texto, por exemplo Portugal pode ser LOCAL ou PESSOA e temos de as destingir ao longo do texto, portanto essas têm de ser diferenciadas para que a informação extraída seja objetiva e com base no especificado na nova TAG.

4.3.1 Experiência 3 - Tentativa de utilização de informação sobre entidades na Coleção Dourada HAREM ReRelEM com a versão 2.0

O objetivo desta versão 2.0 é usar como auxílio o ficheiro XML da Coleção Dourada do Segundo HAREM ReRelEM e com base nesse ficheiro tentar alguma informação sobre a precisão e Recall do sistema. Todas as experiências e melhoramentos anteriores se ligam para obter este sistema para que se proceda a este teste, assim como todo o processo de reestruturação de código para aceitar o documento que se apresenta. Este documento tem a particularidade de incluir a anotação de todas as relações existentes entre entidades em todos os documentos, ou seja, as entidade mencionadas estão devidamente anotadas e corrigidas à mão, o que faz deste um documento que contém um reconhecedor de nomes e entidades ou NER "perfeito" e que pode e vai ser usado no nosso teste.

Este será o último teste e tem como objetivo captar entidades mencionadas por meio de modelos de extração, procedendo a uma extração que foca sobretudo nessas palavras e que, a ser possível, nos deixa especular a eventualidade de continuar o projeto se a investigação for virada para o aperfeiçoamento de um NER interno.

4.3.1.1 Resultados

Após as sucessivas transformações a que o nosso software fora exposto temos agora uma base mais sólida para que este tipo de teste seja analisado. A tabela seguinte mostra um exemplo de como eram as TAGS das entidades mencionadas num normal processo de extração de uma versão anterior e de seguida uma tabela que mostra o processo de transformação para as novas TAGS, na sua versão final:

Entidades Mencionadas e respetiva TAG Penn Treebank:

Reforma_Protestante	Reforma_Protestante	NN	21	pobj
Europa	Europa	VBZ	266	advcl
Avignon	Avignon	DT	53	det
Grande_Cisma	Grande_Cisma	NNP	50	pobj
Portugal	Portugal	null	92	det
Alemanha	Alemanha	JJ	103	prep

Entidades Mencionadas e respetiva TAG de acordo com o ficheiro da Coleção Dourada HAREM:

Reforma_Protestante	Reforma_Protestante	ABSTRACCAO	21	pobj
Europa	Europa	LOCAL	266	advcl
Avignon	Avignon	ACONTECIMENTO	53	det
Grande_Cisma	Grande_Cisma	ACONTECIMENTO	50	pobj
Portugal	Portugal	PESSOA	92	det
Alemanha	Alemanha	LOCAL	103	prep

Agora estas palavras que estão manualmente retificadas no documento da Coleção Dourada do HAREM, são vistas e captadas como entidades mencionadas no texto e podem ser usadas como tal, para assim proceder a uma extração mais objetiva no campo da informação que se pretende realmente extrair. Portanto o que se trata aqui é puramente uma decisão da parte do reconhecedor que injeta a nova TAG ao formato que entra no extrator, este que fará a sua pesquisa de relações e atribuição de modelo para extração de acordo com essa nova TAG, como é explicado de seguida. Apresentado o seguinte modelo de extração para exemplo:

rel arg1 <nsubj< rel:postag=VBZ >dobj> arg2:postag=NN 0.1001

Este modelo de extração que agora se apresenta é um modelo simples que indica o seguinte, primeiro é mostrada a "rel" este é a relação que irá fazer corresponder os seguintes argumentos, a relação que neste caso é um verbo, VBZ, que está ligado por nsubj, sujeito nominal, ao argumento1 e por dobj, objeto direto, ao argumento dois. Este último argumento tem o "postag" que indica que só serão aceites para este modelo palavras com TAGS NN, portanto nomes, por fim o número que limita o modelo representa a confiança para extração.

Com este modelo de extração conseguimos apanhar várias relações, de entre as quais temos as seguintes:

Da análise da extração podemos ver que temos o -arg1 -rel -arg2, assim como o que gera a extração, ou seja, o "Enabler". A possibilidade de gerar novos modelos de extração permite que sejam captadas novas relações, aplicando "postag" como identificador de argumento e verbo. Como falado anteriormente, depois de captar as entidades mencionadas, seria de

Tabela 4.2: Extração v2.0

Extração #1	-Henrique -reafirma -a ortodoxia
Enabler:	Apesar,de uma certa deriva em direção ao luteranismo
Text:	Henrique reafirma a ortodoxia
Extração #2	-A revolta -produz -uma nova forma
Enabler:	normalmente,uma nova forma de pensamento quanto à forma de organização da sociedade . Assim foi com a Reforma_Protestante
Text:	A revolta produz uma nova forma

extrema relevância injetar as TAGS associadas às mesmas no nosso ficheiro de tokenização, por exemplo, ao invés de ter a palavra Reforma_Protestante com TAG NN iremos ter a palavra Reforma_Protestante com a nova TAG ABSTRACCAO|ACONTECIMENTO, esta TAG como substitui a anterior, NN, fora colocada no modelo de extração para captar somente argumentos com essa TAG, como mostrado no próximo exemplo:

Novo modelo de extração com a alteração do TAG Penn Treebank para um novo TAG presente no ficheiro XML que a identifica, EM.

rel arg1 <nsubj< rel:postag=VBZ >dobj> arg2:postag=ABSTRACCAO 0.1001

Com este teste, o resultado esperado seria o seguinte:

Tabela 4.3: Extração v2.0

Extração #1 -conciliação -da -***Reforma_Protestante***
Enabler: - à,Reforma_Protestante do ocidente europeu))
Text: -conciliação da Reforma_Protestante

Portanto, seria possível captar a Reforma_Protestante com o novo TAG.

4.3.1.2 Conclusão

Após extração com o novo modelo apresentado se concluí que tal não deu os resultados esperados, pelo que mesmo após algumas alterações ao modelo de extração não foi apresentado qualquer resultado positivo quando corrido com o mesmo conjunto de palavras que o exemplo anterior, isto indica que o OllIE requer apenas as TAGS do Treebank que usa para extração, ou seja, apenas as TAGS PennTreebank são permitidas como se comprova no seguinte código presente no software OllIE:

```
object OpenParse {
  val LEMMA_BLACKLIST =
  Postagger.simplePrepositions + ‘‘like’’ + ‘‘be’’
  val VALID_ARG_POSTAG =
  Set(‘‘NN’’, ‘‘NNS’’, ‘‘NNP’’, ‘‘NNPS’’, ‘‘JJ’’, ‘‘JJS’’, ‘‘CD’’, ‘‘PRP’’)
```

```
val logger = LoggerFactory.getLogger(this.getClass)

/** An url to the default model */
def defaultModelUrl: URL = {
  val path = ``openparse.model''
  val url = this.getClass.getResource(path)
  require(url !=
  null, ``Default model could not be found: '' + path);
  url
}
```

Cujo se encontra no diretório:

```
bruno@bruno:~/ollie-master/core/src/main/scala/edu/knowitall/openparse$
```

Este problema teve impacto na extração final de informação por meio de entidades reconhecidas, portanto passámos a outra solução que seria tida como final e na qual seria possível esse tipo de extração mais focada para o objetivo a extrair.

4.3.2 Experiência 4 - Avaliação da versão 2.0 usando apenas anotação morfossintática

No sentido de melhorar a versão 2.0, na qual é apresentado o ficheiro da Coleção Dourada que nos permite estes testes, passamos agora a uma modificação, na qual é descartada a possibilidade de uso das suas EM como parte da solução e colocada em evidência o uso de apenas Part of Speech Tags.

Esta solução passa apenas por não alterar as TAGS Penn Treebank usadas para extração e que identificam cada palavra, mas sim no ficheiro de modelos openparse.model, este que já foi explicado e no qual podemos injetar por meio de REGEX e POSTAGS as ordens de extração que queremos encontrar de determinado tipo, como este exemplo que se apresenta:

arg1 <nsubj< rel:postag=VBZ >dobj> arg2:postag=NNP:regex=Henrique_VIII

Captar um determinado nome Henrique_VIII, cujo se encontra com o tagset Penn Treebank "NNP", então para proceder à sua captação temos de proceder à conceção de um modelo que capta o nome próprio, neste caso NNP, mas cujo REGEX dará apenas importância àquele nome que queremos. Vamos proceder ao teste no qual é dado o modelo de extração apresentado para que consiga captar o segundo argumento como tendo o postag "NNP" e o nome Henrique VIII.

4.3.2.1 Resultados

Com este novo sistema já conseguimos ter bem especificadas as nossas entidades mencionadas de modo a proceder a uma extração eficaz por meio de REGEX e Postag.

Tabela 4.4: Resultado da Extração v2.0

Extração #1 -a ortodoxia -reafirma -***Henrique_ VIII***   Enabler: -(EnablingCondition(Apesar,de uma certa deriva em direção ao luteranismo,))   Text: -a ortodoxia reafirma ***Henrique_ VIII***

4.3.3 Experiência 5 - Avaliação usando identificadores únicos da Coleção Dourada

Esta experiência tem como base apenas algumas alterações a nível de processamento do documento de entrada. Com o ficheiro XML do HAREM da Coleção Dourada temos bem expostas as relações que existem entre as entidades mencionadas, foi feita uma pequena alteração ao software de modo a permitir a extração destas, ou seja, com este ficheiro anotado que possui as entidades mencionadas, respetiva identificação e correlação conseguimos extrair essas mesmas relações entre frases.

Com objetivo definido passamos ao método para o teste, com o seguinte exemplo:

Ficheiro XML anotado da Coleção Dourada:

```
(...)foco onde surgiram estes pensamentos.
A imprensa, inventada na <EM ID=‘‘H2-dftre765-9’’
CATEG=‘‘LOCAL’’ TIPO=‘‘HUMANO’’ SUBTIPO=‘‘PAIS’’ COREL=‘‘H2-dftre765-37’’
TIPOREL=‘‘incluido’’>Alemanha</EM>
por <EM ID=‘‘H2-dftre765-10’’ CATEG=‘‘PESSOA’’ TIPO=‘‘INDIVIDUAL’’
COREL=‘‘H2-dftre765-9’’
TIPOREL=‘‘natural_de’’>John Gutenberg</EM>, foi importante (...)
```

Com este exemplo de frase do nosso XML verificamos que a palavra John Gutenberg tem uma correlação, H2-dftre765-9, com a palavra Alemanha cujo o identificador é H2-dftre765-9, por via de um tipo de relação natural_de.

A informação gerada seria o seguinte:

```
John_Gutenberg ; natural_de ; Alemanha
```

4.3.3.1 Resultado

Posto isto, com a nossa nova alteração podemos captar todas as relações existentes entre duas ou mais identidades mencionadas que possuam o identificador igual ao identificador de correlação. Contudo estes estão somente especificados pelas TAGS “ID” e “COREL” de uma dada entidade mencionada, a qual desaparece quando limpas as TAGS XML do texto para análise, perdendo portanto essas mesmas relações entre as entidades.

Só faz sentido captar essas relações se as mesmas existirem no texto original, caso contrário iríamos apenas iterar o texto com um procura exaustiva e isso não seria extração

de informação, só faz sentido se essas relações estiverem presentes no texto, por exemplo, limpa a frase XML dada:

"(...)demasiado distante do foco onde surgiram estes pensamentos. A imprensa, inventada na Alemanha por John_ Gutenberg (...)"

Não conseguimos encontrar em local nenhum a informação que John Gutenberg é natural da Alemanha, contudo podemos extrair a seguinte informação:

Tabela 4.5: Resultado de extração experiência 5

Extração #1- (...) -A imprensa, ***inventada na Alemanha por John Gutenberg*** foi importante na divulgação destas ideias. - As 95_Teses_de_Martinho_Lutero

Concluímos então que a criação de modelos de extração específicos que consigam captar a relação anteriormente exposta são impossíveis pois quando é limpo o texto XML para texto puro não existe nenhum traço de informação que ligue essas duas entidades, porém se existir irá ser extraída no caso de existir o modelo presente no ficheiro openparse.model que a capte.

4.4 Avaliação final com ficheiro de modelos otimizado

4.4.1 Motivação e relevância

Nesta fase de testes e resultados, sabendo que não teríamos qualquer avanço com o ficheiro XML anteriormente falado, conforme fora mostrado, fomos forçados a avançar para uma fase final que iria trabalhar sobretudo com os modelos de openparse para extração de relações conhecidas. A motivação para esta última fase foca na obtenção de resultados num determinado corpus de pequeno tamanho, no qual já se conhecem as relações entre argumentos, sendo que o objetivo será captar algumas dessas relações com novos modelos e depois proceder à extração com esses mesmos modelos num corpus grande, no qual será analisado o resultado de ambos, ou seja, se são captadas algumas relações e como tal obter uma precisão de captura das relações criadas tanto no texto conhecido como no texto da Coleção Dourada do segundo HAREM.

4.4.2 Preparação e informação para a avaliação

De seguida se apresentam tipos de modelos usados e presentes no ficheiro openpase de modelos de extração depois de otimizado para esta fase:

```
template
{rel} {arg1:postag=NNP} <cop< {rel:postag=VBZ} >prep> {arg2:postag=NN} 0.1001
{rel} {arg1:postag=NNP} <cop< {rel:postag=VBZ} >pobj> {arg2:postag=NN} 0.1001
{rel} {arg1:postag=NNP} <nsubj< {rel:postag=VBZ} >pobj> {arg2:postag=NN} 0.1001
{rel} {arg1} <nsubj< {rel:postag=VBZ} >prep> {arg2:postag=NNP} 0.1443
{rel} {arg1} <nsubj< {rel1:postag=VBZ} >xcomp> {rel2:postag=VBG} >dobj> {arg2}
    0.1443
{rel} {prep}  {arg1} <nsubj< {rel:postag=NN} >{prep:regex=prep_(.*)}> {arg2}
    0.9555
{rel} {arg1} <nsubj< {rel:postag=VBD:regex=acenar|(...)|virar} >prep_at> {arg2}
    0.0623
{rel} {arg1} <nsubj< {rel1:postag=VB} >xcomp> {rel2:postag=VBG} >dobj> {arg2}
    0.1473
```

```
{rel} {prep}  {arg1} >appos> {rel:postag=NN} >{prep:regex=prep_(.*)}> {arg2}
    0.4031
{rel} em {arg1} <nsubjpass< {rel:postag=VBN} >prep_of> {arg2} 0.0061
{rel} {prep}  {arg1} <nsubj< {rel:postag=VBZ} >dobj> {slot0:postag=NN}
    >{prep:regex=prep_(.*)}> {arg2} 0.0061
{rel} {arg1} <nsubj< {rel1} >xcomp> {rel2} >dobj> {arg2} 0.1443
{rel} {arg1} <nsubj< {rel1:postag=VB} >xcomp> {rel2} >dobj> {arg2} 0.1443
{rel} {arg1:postag=NN} <dobj< {rel:postag=IN} >prep> {arg2:postag=NNP} 0.1443
```

Para execução deste tipo de teste precisamos de um conjunto de relações, essas foram retiradas dos exemplos de tipo de relação presente nas entidades mencionadas do ficheiro XML da CD Harem, no total de 32 relações, das quais foram eliminadas todas as que não fariam sentido para o nosso teste, ou porque eram de extrema complexidade ou porque ligavam a outras relações formando mais que uma, acabando por ficar com um conjunto de 19 relações, destas apenas para exemplo foram escolhidas aleatoriamente as que se seguem.

-Relação-	-#Parágrafos-
Obra *	4
Natural *	2
Autor *	5
Produzido *	2

Depois de obter a lista de relações nas quais vamos focar, o próximo passo será obter texto que contenha essas relações de alguma forma, para tal foi usado o sitio da Linguateca que contém o corpus do CETEMPúblico do qual serão retirados alguns parágrafos que interessam neste teste, ou seja, parágrafos que contenham algum destes tipo de relações apresentadas, formando um total de 13 parágrafos divididos entre as relações.

4.4.3 Método para avaliação

O teste vai ser captado da seguinte forma, depois de criados os modelos para extração, vamos correr cada um dos parágrafos do texto conhecido no software, este dará resultados que serão guardados, como o número de extrações e dos quais são viáveis para o nosso teste. De seguida é feito o mesmo processo, mas para a CD HAREM, onde serão captadas algumas extrações obtidas viáveis e válidas com as relações procuradas, para com isso se concluir a precisão do nosso modelo. Neste tipo de testes, por termos um corpus muito extenso, como é o caso da CD HAREM, será confinada aos cinco melhores modelos e para efeitos de validade, basta encontrar um pequeno número de extrações válidas, se estas existirem no texto original. Obviamente que não serão captadas grandes relações no texto pelas razões já apresentadas.

Apresentação dos cinco (5) modelos usados que possibilitaram a extração:

```
R1 {rel} {arg1} <nsubj< {rel:postag=VBZ} >ccomp> {arg2:postag=NN} 0.1521
R2 {rel} {arg1} <nsubj< {rel} >ccomp> {arg2:postag=NN} 0.1431
```

```
R3 {rel} {arg1} >{prep:regex=(.*)}> {rel} >{prep}> {arg2} 0.4131
R4 {rel} {arg1} <nsubj< {rel:postag=VBZ} >dobj> {arg2} 0.1491
R5 {rel} {arg1} <nsubj< {rel:postag=VBZ} <xcomp< {arg2} 0.1501
```

4.4.4 Resultados - Texto conhecido de relações

Com os parágrafos escolhidos para teste devidamente diferenciados, podemos proceder à extração com o novos modelos de extração que foram feitos e se apresentaram na subsecção anterior para conseguir captar essas relações, sendo que esse modelo será usado para extração de informação no texto da CD HAREM, onde posteriormente com esses resultados teremos a possibilidade de averiguar o que é captado nesse texto e o que fora captado no texto conhecido com o mesmo modelo e com isso obter uma precisão de extração. Junto se apresentam resultados mais marcantes, estes que serão expostos com o texto original e de seguida extração conseguida. com o seguinte formato: {arg1} + {R#} {rel} + {arg2}, no qual {R#} representa o número do respetivo modelo que captou essa relação. Dada a enormidade do corpus que é a CD HAREM não ficou claro o número de extrações de cada relação pelo simples facto de algumas serem muito parecidas a outras, quererem dizer o mesmo, que não faziam sentido e pelo número de extrações no geral ser enorme. Focámos nas mais importantes e relevantes.

4.4.5 Exemplos de extrações obtidos

De seguida se mostram os exemplos das extrações obtidas, inicialmente se apresenta o texto e a extração originada do mesmo.

Obra *

> Neste sentido, «Os Olhos da Ásia» é uma obra, já o disse o realizador, «sobre a intolerância».

Tabela 4.6: Extração com o modelo R1

Extração#1-«Os Olhos a Ásia» -Já o disse o realizador, «sobre a intolerância» é - uma obra
Text: - «Os Olhos a Ásia» é uma obra

O importante é o conteúdo que a obra transmite», sublinha .

Tabela 4.7: Extração com o modelo R2

Extração #1	-O importante - é -o conteúdo
Text:	- O importante é o conteúdo

É um olhar breve sobre a obra de um dos mais fascinantes autores europeus contemporâneos, que encontrou no registo a preto e branco uma belíssima forma de expressar sentimentos e construir atmosferas .

Tabela 4.8: Extração com o modelo R3

Extração #1 -a obra - encontrou no registo a preto e branco -uma belíssima forma
Text: - a obra encontrou no registo a preto e branco uma belíssima forma
Extração #2 -uma belíssima forma - expressar -sentimentos
Text: - uma belíssima forma expressar sentimentos

Finamente, na sexta-feira, o programa encerra com uma intervenção do próprio Graça Moura, que comentará a sua obra e a sua relação com o Porto, seguindo-se um recital de piano por Sofia Lourenço, que interpretará algumas das composições preferidas do homenageado .

Tabela 4.9: Extração com o modelo R3

Extração #1 -na sexta-feira -o programa encerra com uma intervenção do encerra -próprio Graça Text: - na sexta-feira encerra próprio Graça Extração #2 -próprio Graça -um recital de piano por Sofia comentará Lourenço que interpretará algumas das composições preferidas do homenageado -a sua obra Text: - próprio Graça comentará Lourenço , que interpretará algumas das composições preferidas do homenageado a sua obra

---

Natural *

Este pensador natural de Bordéus, e o Salão do Livro desta cidade homenageou-o.

Tabela 4.10: Extração com o modelo R3

Extração #1	-o Salão - desta cidade homenageou-o -Livro
Text:	-o Salão desta cidade homenageou-o Livro

Bebiana Pereira da Gama, natural de Luanda, que é tão portuguesa como Savimbi .

Tabela 4.11: Extração com o modelo R3

Extração #1	-Bebiana - natural -Luanda
Text:	-Bebiana R3 natural Luanda

Autor*

Sólida construção, preparada para resistir aos sismos mais fortes, com um enorme telhado móvel, a mesquita, segundo Michel Pinseau, o arquitecto francês autor do projecto, «ultrapassa, em dimensão, todas as catedrais conhecidas.

Mas, enquanto romancista, Mauriac era, desde a II Guerra Mundial e apesar do Nobel concedido em 1952, considerado o autor de uma obra de temática historicamente muito conotada e, por isso, inteiramente «ultrapassado» .

Tabela 4.12: Extração com o modelo R4

Extração #1	-um enorme telhado - segundo -Michel
Text:	-um enorme telhado segundo Michel
Extração #2	-Sólida construção -dimensão «ultrapassa -todas as catedrais
Text:	-Sólida construção R4 «ultrapassa todas as catedrais

Tabela 4.13: Extração com o modelo R3

Extração #1	-o autor - uma obra -temática
Text:	-o autor uma obra temática

O autor já regressou a Itália, onde vive, mas os seus desenhos continuam expostos ao público até ao final do corrente mês .

Tabela 4.14: Extração com o modelo R4

Extração #1	-O autor -já regressou -a Itália
Text:	-O autor já regressou a Itália

Os direitos de autor pertencem a um poeta chinês chamado Li Wei, que Zink conheceu num desses «laboratórios» de escrita .

Tabela 4.15: Extração com o modelo R3

Extração #1	-um poeta - que conheceu -Zink
Text:	-um poeta que conheceu Zink

Ferreira de Castro e o Brasil e da influência do país adoptivo no autor de «A Selva» .

Tabela 4.16: Extração com o modelo R3

Extração #1	-Ferreira - país adoptivo no autor -«A Selva»
Text:	-Ferreira país adoptivo no autor «A Selva»

Produzido*

Um trabalho de J. Alves de Sá produzido na fábrica da Viúva Lamego em 1940

Tabela 4.17: Extração com o modelo R3

Extração #1	-Um trabalho - na fábrica -Lamego
Text:	-Um trabalho na fábrica Lamego

O primeiro disco conceptual produzido em Portugal na área da música pop tem a assinatura de José Cid e chama-se Dez mil anos depois entre Vénus e Marte .

Tabela 4.18: Extração com o modelo R3

Extração #1	-O primeiro disco - tem a assinatura de José Cid -na área
Text:	-O primeiro disco tem a assinatura de José Cid na área

4.4.6 Precisão e Recall

Para obter uma marca de performance do software com este teste, vamos escolher uma aproximação ao texto a ser processado. Estes testes são apresentados de seguida, com as entidades mencionadas devidamente concatenadas com o caracter "_" de forma a que sejam vistas como uma só palavra, que pode ou não ser reconhecida pelo tokenizer ao ser atribuída a tag referente, e uma outra versão que visa o texto puramente limpo em que não se indicam entidades mencionadas. Se segue o seguinte exemplo.

Com entidades mencionadas concatenadas por via de caracter "_", o tokenizer reconhece e aplica a TAG, neste caso, NNP, etiquetando como num nome próprio.

```
1  Assim Assim RB RB 2 advcl
2  foi  foi  VBZ  VBZ  0  cop
3  com  com  IN IN 2 ccomp
4  a  a  DT DT 5 det
5  Reforma\_Protestante Reforma\_Protestante NNP NNP 3 pobj
6  .  .  PUNCT PUNCT 2 punct
```

Entidades mencionadas separadas, em que "Reforma" e "Protestante" são vistas como dependente e portanto se relacionam melhor no extrator.

```
1  Assim Assim RB RB 2 advcl
2  foi  foi  VBZ  VBZ  0  cop
```

```
3  com   com   IN IN 2 ccomp
4  a  a  DT DT 5 det
5  Reforma Reforma NN NN 3 pobj
6  Protestante Protestante NNP NNP  5  prep
7  .  .  PUNCT PUNCT 2 punct
```

Conseguimos ver à partida nas tabelas que se seguem que o texto simplesmente limpo de TAGS XML e cujas entidades mencionadas não sofreram qualquer processamento de edição, se obtêm resultados mais precisos com 815 mais extrações, portanto uma extração mais profunda, isto deve-se sobretudo às dependências associadas, pois o software é perspicaz o suficiente para encontrar grande parte dessas entidades e juntar as mesmas de acordo com as suas dependências.

Para teste final e achar a precisão do nosso software, vamos testar os modelos já descritos e usados no exemplo apresentado, mas desta vez em todo o texto da CD Harem.

Para avaliação da precisão, vamos procurar as existências de cada uma das relações estudadas no nosso corpus da CD Harém, para que se conclua se existiu extração e de quantas frases essa extração está bem formada. Entenda-se que com "extração bem formada" se quer dizer que a extração faz sentido ao utilizador sem que este receba o texto original e que sobretudo possua a nossa relação procurada. Só desse modo será possível afirmar que temos um modelo de extração verdadeiramente eficaz para que se proceda à extração de informação do corpus total da CD do HAREM. Este teste foi executado uma e uma só vez, portanto para calcular a precisão ou alcance vamos somente contabilizar cada uma das relações extraídas no total de extrações numa primeira fase e proceder a uma análise desses mesmos resultados.

-#Extrações com EM's concatenadas-	-#Extrações com EM's separadas-
9588	10403

Dados os resultados de extração vamos somente analisar as extrações com as Entidades Mencionadas separadas e cruzar essa informação com a nossa tabela de relações de procura.

-Relação-	-# aproximado de Extrações da CD HAREM-
Obra *	114
Natural *	81
Autor *	101
Produzido *	155

No campo de relações encontradas, conseguimos provar que o nosso modelo consegue, dado um grande corpus, captar relações presentes na lista de relações estudada, como se consegue provar nos seguintes exemplos dados da CD HAREM. Ainda que estes exemplos não sejam os melhores, temos consciência de que o texto presente da CD não prima pela estrutura narrativa, pelo que é possível que alguns dos exemplos não façam grande sentido, também o tamanho do corpus dificulta não só cálculos de precisão do software mas também textos que originem estas extrações. No entanto foram escolhidos alguns que se

apresentavam mais completos de entre os inúmeros resultados e extração antes de proceder a um teste com um espaço amostral reduzido.

Obra *

> “ (...)uma melhor compreensão do trabalho de Carlos Gomes , cuja obra de maior sucesso, “ O Guarani ”, estreou no Teatro Scala de Milão (...)”

> “(...)O Superflex ainda distribuiu à imprensa, no dia da coletiva, o documento A obra de arte que os brasileiros não terão permissão de ver na Bienal (...)”

Tabela 4.19: Extração da relação “Obra *”

Extração #1	-Carlos Gomes - cuja obra -maior sucesso
Text:	-Carlos Gomes cuja obra maior sucesso
Extração #2	-O Superflex -distribuiu à imprensa , no dia da coletiva distribuiu o documento “ A obra de arte -imprensa
Text:	-O Superflex distribuiu o documento “ A obra de arte imprensa

---

Natural *

> “(...)crítico musical do « Diário de Notícias », Joaquim de Seabra Pessoa (38), natural de Lisboa (...)”

> “(...)Nogueira Pessoa (26), natural da Ilha Terceira (Açores)(...)”

Tabela 4.20: Extração da relação “Natural *”

Extração #1	-Joaquim de Seabra Pessoa -natural de -Lisboa
Text:	-Joaquim de Seabra Pessoa natural Lisboa
Extração #2	-Nogueira Pessoa - natural da -Ilha
Text:	-Nogueira Pessoa natural da Ilha
Extração #3	-Nogueira Pessoa - natural da -Terceira
Text:	-Nogueira Pessoa natural da Terceira

Autor *

"(...)excelente professor de latim, entusiasta do ensino clássico, autor de livros na suas especialidade e um grande propagandista do magistério do latim entre seus alunos (...)"

"(...)Rocha Lima, Matos Peixoto e o autor destas linhas, se reuniram para alugarem uma sala no centro da cidade(...)"

"(...)Aproveitando a maré boa, Gaines adaptou as histórias do conhecido autor Ray Bradbury e jogou no mercado Weird Science e Weird Fantasy(...)"

Tabela 4.21: Extração da relação "Autor *"

Extração #1 -excelente professor - autor -livros
Text: -excelente professor autor livros
Extração #2 -Rocha Lima -o autor -destas
Text: -Rocha Lima o autor destas
Extração #3 -Gaines -a maré boa adaptou as histórias do conhecido autor Ray Bradbury -publicações
Text: -Gaines adaptou as histórias do conhecido autor Ray Bradbury publicações

---

Produzido *

"A revolta histórica produz normalmente uma nova forma de pensamento quanto à forma de organização da sociedade."

"Tal documento contém referências à actuação da protecção civil nos contextos de crises humanitárias, pelo que foi também discutido no Grupo de Trabalho PROCIV , tendo sido produzido um aconselhamento em Julho."

Tabela 4.22: Extração da relação "Autor *"

Extração #1 -Fatores -A revolta histórica produz normalmente produz uma nova forma de pensamento quanto à - revolta
Text: -Fatores produz uma nova forma de pensamento quanto à revolta
Extração #2 -Tal documento -sido produzido um aconselhamento em_Julho contém pelo que foi também discutido no Grupo de Trabalho PROCIV -referências
Text: -Tal documento contém pelo que foi também discutido no Grupo de Trabalho PROCIV referências

---

Conforme esperado algumas relações, embora captadas, estariam mal formadas ou sem sentido, contudo se admite que o tamanho do modelo de extrações seria pequeno demais para alargar essa pesquisa, no entanto e com os resultados aqui apresentados se prova que se conseguiu encontrar algumas relações apenas com o modelo de relações estudado e com mais e melhores modelos se poderá atingir uma melhor extração.

Também algum trabalho para descartar as relações que se complementavam deveria ser tomado em conta, pois é fácil perceber que existem extrações que nos apresentam basicamente a mesma informação. Os resultados que se seguem mostram a percentagem de ocorrências de cada uma das relações estudada no total do corpus, assume-se que estes resultados poderiam ser melhores se fosse alargada a pesquisa a estudar com sinónimos, portanto quando se indica a relação está-se no fundo a mostrar quantas extrações relacionam dois argumentos por via dessa mesma relação.

-Relação-	-% de ocorrências positivas sobre o total de extrações-
Obra *	1.10%
Natural *	0.78%
Autor *	0.97%
Produzido *	1.49%

Visto termos uma largo conjunto de extrações, 10403 para ser mais preciso, para ter algum valor de precisão do software vai ser feito o seguinte teste, escolhidas as duzentas, 200, primeiras extrações produzidas pelo sistema e de seguida analisar esse conjunto de extrações escolhidas e contabilizar quantas destas estão dentro da gama de relações procuradas. Espera-se um valor baixo conforme já fora explicado.

-Relação-	-# de ocorrências positivas sobre 200 extrações-
Obra *	3%
Natural *	0%
Autor *	6%
Produzido *	3%

Tiradas as 200 primeiras extrações que foram iteradas com o objetivo de encontrar e contabilizar quantas se inserem na nossa tabela de relações procuradas, nesta situação foram contabilizadas também as extrações que possuíam sinónimos da lista de relações, pois o número de relações que se inseriam na nossa lista seriam nulos ou quase nulos.

Para recall de software a analise a ser feita será com base nas relações anotadas manualmente no ficheiro CD HAREM e através dessas encontrar relações nas nossas extrações. Para este teste, vamos mais uma vez utilizar as 200 primeiras extrações e procurar alguma semelhança com a nossa lista de relações e as relações anotadas manualmente presentes no texto CD HAREM.

Vamos tomar como exemplo na qualidade de "Produzido *" a seguinte extração: A Revolta - produz - uma nova forma de pensamento.

No texto CD HAREM a relação do pedaço de texto que originou a extração contém o seguinte tipo de relação: “ TIPO=“IDEIA|EFEMERIDE” ”. Este exemplo mostra a razão pelas quais não são encontradas extrações que façam algum tipo de combinação com estas correlações sobre entidades mencionadas.

Entre outros exemplos analisamos que, mais uma vez, é impossível obter um recall do software com o texto da CD HAREM, pois além das suas relações e correlações não possuírem qualquer ligação com as relações presentes na tabela a ser analisada, sabemos que esses tipos de relação manualmente anotados são eliminados na limpeza XML do texto, portanto perde-se qualquer tipo de referência a essa relação.

Por fim, após analise sobre o texto da CD HAREM encontramos algumas ocorrências das relações estudadas que não foram captadas pelo extrator, esse facto deve-se sobretudo à etiquetação de cada token e consequentemente das suas dependências, pois basta que uma não indique continuidade de relação com dependências entre palavras para que a extração não seja válida. Mesmo com esse erro em causa, podemos afirmar a positividade dos resultados nos termos de análise de extração de informação per si. Porém na análise segundo a CD HAREM no que diz respeito às Entidades Mencionadas e manualmente anotadas no texto, nada podemos concluir, pois estas apresentam-se um tanto quanto incongruentes no ponto de vista lógico, pelo que nada se pode relacionar com as extrações efetuadas conforme já fora estudado.

Capítulo 5

Conclusões

Neste último Capítulo se apresenta a conclusão sobre o trabalho realizado, aqui vamos tratar de expor um resumo, alguns problemas encontrados e soluções para os mesmos, as diversas fases passadas, principais resultados, evolução para o sistema desenvolvido e futuro da EI.

5.1 Resumo do trabalho realizado

Este projeto passou por diversas fases, algumas das quais abandonadas devido à falta de resultados positivos. Na versão de aplicação de teste, em que se trabalha sobretudo com o freeling e corpus de entrada para extração de um ficheiro ConLL viável para a próxima fase, foi sobretudo relevante a extração do ficheiro em si, ou seja, o facto de podermos criar um ficheiro através de um corpus português, cujo seria etiquetado com o treebank EAGLES que está diretamente associado à língua Portuguesa, seria um grande avanço na continuação do projeto, pelo que foi um ponto marcante. Não só a extração deste mesmo ficheiro como também modelos de chunking, PoS e dependências.

Abandonada a primeira fase, na segunda versão temos um software que tem a particularidade de possuir módulos e como tal ser programado por medida, sendo que aqui a medida seria a inclusão de ficheiros binários, já tratados para lidar com a língua portuguesa, portanto binários cedidos pelo projeto OpenNLP, o qual contém esses mesmo modelos e que nos foram extremamente úteis. Os resultados de extração a partir dessa versão foram o culminar de uma pesquisa exaustiva relativamente ao software para prosseguir o trabalho.

Após este processo de testes sobre soluções que podiam ser usadas foi escolhido o software com o qual se iria trabalhar, como a seguir se apresenta. O trabalho foi iniciado com o OllIE, após uma fase mais crítica que passou pela escolha do software de extração a ser usado e a sua conjugação para português conforme exposto, tendo em conta a dificuldade em encontrar estado da arte para este formato. A escolha do OllIE fora fulcral para o avanço da matéria, contudo este apresentava o problema já esperado, como extrair informação em Português. Para fazer face a esse problema o software complementa-se com o MaltParser que consequentemente fornece o ficheiro .mco para reconhecimento da língua Portuguesa.

O software programado resolve todos os problemas de compatibilidade, trabalhando com as API's de cada uma das soluções e conjugando-as em formato pipeline.

Visto que o OllIE aceita somente tagset Penn Treebank esse problema fora solucionado com a combinação dos Treebanks associados, o Treebank da floresta sintática cuja é atribuída aos tokens no momento da tokenização e o Treebank do Ollie que é Penn Treebank, sendo que este mapeamento tornou a extração possível. Finalmente todo o trabalho de lapidação de resultados fora solucionado com metodologias no código de extração e criação de novos modelos de extração fiáveis para os nossos objetivos.

Passando para a versão 2.0 do sistema, cujo visou sobretudo numa alteração das regras que modelam a extração, estas que foram alvo de uma tradução exaustiva, que se verificou inútil não apresentando uma melhor extração dados os avanços tomados, contudo, foram aprendidas as formas de lidar com essas mesmas regras e a alteração das mesmas para captar mais relações, também alterações no código possibilitaram uma melhor extração, no que diz respeito ao que despoleta a extração, como é o caso do "Enabler" que fora adicionado.

O problema inicial desta versão mostra o avanço que existiu, não só no método de aproximação, como também nos resultados, estas versões foram sempre sendo melhoradas e alteradas com vista a produzir resultados úteis para futuros trabalhos. Nesta fase final, foi implementada a versão 2.0, que foi sendo alvo de algumas experiências e testes que trabalham um melhoramento dos modelos de extrações, assim como novos documentos de entrada.

Cada sistema está avaliado consoante o grau de satisfação obtido pelas extrações, no primeiro sistema 1.0 a avaliação passou pela simples extração de noticias e suas relações através dos modelos de relação simplesmente traduzidos, o qual apresentava algum atraso na extração dada a sua quantidade, algo que fora melhorado de seguida, em que o ficheiro de modelos fora alterado de uma forma geral, o que produziu resultados positivos.

Entramos agora na fase da versão 2.0, ou final, esta versão joga com vários fatores, um deles é o facto de se proceder a testes que contabilizem o esforço induzido neste projeto, portanto uma precisão de extração sobre determinados corpus. Esta versão foca na Coleção Dourada Segundo HAREM ReRelEM, sendo este um ficheiro de formato XML com um corpus vasto, manualmente anotado.

Aqui as aproximações focaram não só no melhoramento geral do software mas também em obter precisão de acordo com os modelos escolhidos, dando como documento de análise o texto da CD HAREM no qual se mostra que o ficheiro embora seja manualmente anotado, dessas anotações em nada estão de acordo com a narrativa apresentada, ainda assim se conseguem algumas extrações interessantes.

Ao longo destas várias fases foram debatidos problemas como a falta de documentação em algumas das soluções, outputs de software que não eram compatíveis entre si e portanto não seriam expostos a pipeline de informação, como por exemplo ficheiros ConLL de diferentes tipos, com diferentes campos ou com campos nulos, versões de MaltParser que não se conjugavam com os dados de entrada, neste caso os ficheiros ConLL para testes, a migração do software FreeLing para o Linux cujo apresentava diversos problemas no campo do sistema operativo e de formatos de tokenização como as TAGS e respetivo Treebank.

5.2 Principais resultados

O resultado principal que mais caracteriza esta dissertação de mestrado é, não só uma modelação principal no que diz respeito ao uso do OllIE para extração de informação sob relação aberta para Português, mas sobretudo um avanço e passo efetuado nesse mesmo campo, existindo muito pouco sobre o tema em background de EI para português. Este facto representava uma dificuldade inerente à linguística, devido a todas as possíveis conotações existentes em português, contudo foi feito o possível para que tal não representasse um problema crítico.

Os resultados principais podem ser resumidos à seguinte lista:

1. Desenvolvimento do pipeline completo para Português que fornece informação para a obtenção de extração de informação

2. Integração do MaltParser no software para que fosse reconhecida a língua Portuguesa.

3. Identificação dos limites em termos de regras impostos pelo OllIE, sem que para isso tenha sido levada a cabo uma alteração no código interno deste.

4. Tradução inicial das regras de extração de Inglês para Português e escolha das mais relevantes.

5. Conjunto de regras de extração desenvolvidas manualmente para novos modelos de extração no ficheiro openparse.

6. Tentativas de obtenção de resultados e avaliação, em especial com corpus HAREM.

5.3 Sugestões de continuidade

O trabalho descrito está longe de terminado, portanto consideramos necessário dar-lhe continuidade explorando os seguintes aspetos: É importante evoluir os módulos do sistema desenvolvido e com isso proceder a uma melhor análise dos dados de entrada, como é o caso do OllIE e alterações fundamentais no Treebank associado. No entanto, analisar evoluções recentes poderá ser um novo ponto de partida como se segue explicado, considerar novas ferramentas com especial foco para o módulo responsável pela extração propriamente dita será uma aproximação correta. Não podemos descurar a criação de um novo sistema incluindo suporte para Português nativamente.

5.3.1 Evolução do sistema desenvolvido

É fator chave alterar o ficheiro Scala que rege a entrada do Treebank e faz a asserção do mesmo no OllIE para que sejam adicionados novos TAGS e com isso poder testar se é possível proceder à extração por modelação através de postag e achar as TAGS de

entidades conhecidas devidamente anotadas, para que com esse feito, e caso seja positivo, poder permitir extrações que usem um NER, como fora tentado neste projeto.

Também o melhoramento do ficheiro que contém os modelos de extração, openparse, é algo decisivo de se melhorar para que se captem mais e melhores relações.

Uma especial atenção ao conjunto de TAGS da floresta sintática e qual a TAG Penn Treebank que a caracteriza melhor.

Para um trabalho futuro, com uma implementação sob big data, a performance do nosso software de extração pode ser melhorada, como por exemplo, a adição de threads independentes que estão à escuta de parágrafos do texto e que ordenadamente as processa, aqui não só conseguiríamos um uso exaustivo do software como também seria permitida a escalabilidade do mesmo, tomando à partida que o único bottleneck seria apenas o troughput do extrator, o que representa escalabilidade do nosso programa, ou seja, vamos ter quantos parágrafos quanto for o número de processadores a recebê-los.

5.3.2 Adoção de outras ferramentas

Já após a entrega da primeira versão desta Dissertação tivemos conhecimento de uma publicação da revista Linguamática de Junho de 2017, que faz referência a um toolkit ou suite multilingue extremamente interessante não só para o futuro da análise linguística mas sobretudo para extração de informação. O LinguaKit[19] é uma ferramenta programada em Perl, de código aberto e com variedades linguísticas como o Português, Espanhol, Inglês e Galego para processamento de linguagem natural que contém módulos já apresentados nesta dissertação, como etiquetação, PoS tagging, análise sintática entre outras. Contudo esta ferramenta vai mais além e processa análise de sentimentos ou minaria de opiniões, identificador e classificador de entidades, extração de termos multipalavra, palavras chave, concordâncias, identificação de língua, correção e avaliação linguística e extração de informação ou sumarização.

LinguaKit está disponível como serviço web e acessível a todos por via de um limite de usos diários dividido em módulos independentes entre si, o que é relativamente bom para projetos com o que aqui se apresentou, cujos estão divididos em quatro categorias com os seguintes módulos que se apresentam:

- Análise básica - Conjugador verbal, segmentador de orações, tokenizador e splitter
- Análise profunda - Lematizador PoS-tagger, identificador de entidades (NER), classificador de entidades (NEC), identificador de correferência e analisador sintático em dependências
- Sistemas de extração - Palavras chave, expressões multipalavra, análise de sentimento/opinião e relações semânticas (open IE)
- Aplicações linguísticas - Sumarização, anotação semântica (com EL), concordâncias (palavras chave em contexto), identificação de línguas correção/avaliação e linguística (léxica e gramatical)

Os principais módulos de LinguaKit foram desenhados e implementados nos últimos cinco anos, encontramos a maior parte deles descritos em diferentes publicações sendo que alguns deles são utilizados no nosso trabalho. Os primeiros módulos realizam um pré-processamento do texto, de seguida realizam identificação de fronteiras e orações com base em máquinas de estado finitas e listas de abreviaturas que terminam com pontuação. Também tokenização e splitting são utilizados, para separação e atribuição de lemas a cada token.

Numa breve análise ao módulo extrator de relações, podemos ver algumas semelhanças com o apresentado neste projeto de dissertação, na qual se obtém um conjuntos de relações entre objetos, portanto objeto1, relação, objeto2, selecionadas por um sistema de extração de informação aberta. Esse sistema está baseado por regras e tem como entrada um texto analisado em dependências em formato ConLL.

De entre os módulos já descritos utilizados pelo LinguaKit, existem alguns que não foram utilizados no nosso trabalho, mas que formariam uma solução mais rica caso fossem, de seguida se apresentam alguns.

Resolução de correferência a nível de entidade

Este módulo utiliza um texto com as entidades mencionadas disponíveis através de um NER e devidamente classificadas e aplica filtros cujos atribuem um identificador numérico a cada uma das ocorrências. Idealmente este identificador será igual para cada uma das menções que registam a mesma atividade, caso tal não se verifique, o algoritmo aplicado pela resolução de correferências irá tentar corrigir erros prévios da classificação semântica.

Análise de sentimentos

Este sistema de análise de sentimentos classifica cada oração como tendo uma opinião positiva, negativa ou neutra. O núcleo desde módulo é um classificador bayesiano, que é um conceito de probabilidade que assume uma hipótese caso não haja uma certeza se é positiva, negativa, ou neutra. Este é treinado com texto pré anotado com opiniões definidas, que também utiliza um léxico de polaridade e regras sintáticas para identificação de marcadores linguísticos que intensificam ou mudam a polaridade das palavras.

Anotação e ligação semântica

Este módulo identifica termos relevantes do texto e liga-os a conceitos presentes em bases de dados externas como DBpedia. Esta tarefa consiste em relacionar termos mencionados em texto e os conceitos de uma base de dados ontológica, normalmente conhecida como ligação de entidades.

Corretor linguístico

Desenvolvido principalmente em galego, o sistema identifica e classifica diferentes tipos de erros habituais, tanto no léxico como no gramatical. Existe contudo versões básicas

para português e espanhol, mas estas precisam de ser trabalhadas.

O LinguaKit seria uma ótima ferramenta para ajuda no desenvolvimento do projeto apresentado nesta dissertação, o qual seria certamente mais rico em termos de EI. O pacote LinguaKit tem a particularidade de ser de fácil acesso e com módulos de análise linguística diversos. Contudo ainda existe um caminho a percorrer com o objetivo de continuidade do mesmo com vista a melhoramentos no desempenho e ampliação de alguns módulos, assim como enriquecer-se com funcionalidades simples mas úteis para linguistas investigadores.

5.3.3 Desenvolvimento de um sistema para Português

Pesquisa e desenvolvimento de novas metodologias de extração, não só a utilização de um extrator desenvolvido puramente para o Inglês que fora alterado para o funcionamento com Português, mas sim um novo software que de forma nativa funcione com o Português, tomando como estado da arte o projeto que aqui se apresenta. Concluímos assim que o nosso software poderá ser o início de um grande passo na Extração de Informação para a língua Portuguesa e a motivação para um extrator puramente para Português.

Bibliografia

[1] Named-entity recognition. `https://en.wikipedia.org/wiki/Named-entity_recognition`.

[2] Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Polyglot: Distributed word representations for multilingual nlp. *arXiv preprint arXiv:1307.1662*, 2013.

[3] Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *IJCAI*, volume 7, pages 2670–2676, 2007.

[4] Steven Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[5] António Branco and João Silva. Evaluating solutions for the rapid development of state-of-the-art pos taggers for portuguese. In Maria Teresa Lino, Maria Francisca Xavier, Fátima Ferreira, Rute Costa, and Raquel Silva, editors, *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC2004)*, pages 507–510, Paris.

[6] Eric Brill. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language*, pages 112–116. Association for Computational Linguistics, 1992.

[7] Sabine Buchholz and Erwin Marsi. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pages 149–164. Association for Computational Linguistics, 2006.

[8] Paula Carvalho, Hugo Gonçalo Oliveira, Diana Santos, Cláudia Freitas, and Cristina Mota. Segundo harem: Modelo geral, novidades e avaliação. *quot; In Cristina Mota; Diana Santos (ed) Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM Linguateca 2008*, 2008.

[9] Michael Collins, Nigel Duffy, et al. Convolution kernels for natural language. In *NIPS*, volume 14, pages 625–632, 2001.

[10] Cunningham, Hamish, Maynard, Diana, Bontcheva, Kalina, Tablan, and Valentin. A framework and graphical development environment for robust nlp tools and applications. In *ACL*, pages 168–175, 2002.

[11] Hamish Cunningham, Yorick Wilks, and Robert J Gaizauskas. Gate: a general architecture for text engineering. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 1057–1060. Association for Computational Linguistics, 1996.

[12] Cutting, Doug, Kupiec, Julian, Pedersen, Jan, Sibun, and Penelope. A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, ANLC '92, pages 133–140, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics.

[13] Domingues, Miriam Lúcia, Favero, Eloi Luiz, de Medeiros, and Ivo Paixão. O desenvolvimento de um etiquetador morfossintático com alta acuracia para o português. *Avanços da Linguística de Corpus no Brasil. São Paulo: Humanitas*, pages 267–286, 2008.

[14] Eleutério, Samuel, Ranchhod, Elisabete, Mota, Cristina, Carvalho, and Paula. Dicionarios electronicos do português. caracteristicas e aplicações. In *Actas del VIII Simposio Internacional de Comunicacion Social*, pages 636–642, 2003.

[15] Oren Etzioni, Robert E Bart, Michael D Schmitz, Stephen G Soderland, et al. Open language learning for information extraction, November 18 2013. US Patent App. 14/083,342.

[16] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics, 2011.

[17] Claudia Freitas, Paulo Rocha, and Eckhard Bick. Um mundo novo na floresta sintá (c) tica–o treebank do português. *Calidoscópio*, 6(3):142–148, 2008.

[18] Pablo Gamallo and Marcos Garcia. Freeling e treetagger: um estudo comparativo no âmbito do português. Technical report, Technical report, Universidade de Santiago de Compostela, 2013.

[19] Pablo Gamallo and Marcos Garcia. Linguakit: uma ferramenta multilingue para a análise linguística e a extração de informação. 9(1):19–28, July 2017.

[20] Marcos Garcia and Pablo Gamallo. Análise morfossintáctica para português europeu e galego: Problemas, solucoes e avaliacao. *Linguamática*, 2(2):59–67, 2010.

[21] Júnior, Carlos Mendonça, Macedo, Hendrik, Bispo, Thiago, Santos, Flávio, Silva, Nayara, Barbosa, and Luciano. Paramopama: a brazilian-portuguese corpus for named entity recognition. *Encontro Nac. de Int. Artificial e Computacional*, 2015.

[22] Leech, G, Wilson, and A. Recommendations for the morphosyntactic annotation of corpora eagles report, 1996.

[23] GNU General Public License. Gnu general public license. *Retrieved December*, 25:2014, 1989.

[24] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.

[25] Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The penn treebank: annotating predicate argument structure. In *Proceedings of the workshop on Human Language Technology*, pages 114–119. Association for Computational Linguistics, 1994.

[26] De Marneffe, Marie-Catherine, MacCartney, Bill, Christopher D Manning, et al. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454. Genoa, 2006.

[27] André Martins. TurboParser - Dependency Parser with Linear Programming. `http://www.cs.cmu.edu/~ark/TurboParser/`.

[28] Lluís Màrquez. Semantic role labeling: An introduction to the special issue - computational linguistics. In *Semantic Role Labeling: An Introduction to the Special Issue.*, 2008.

[29] Joakim Nivre, Johan Hall, and Jens Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC*, volume 6, pages 2216–2219, 2006.

[30] Apache OpenNLP. Apache software foundation. *URL http://opennlp. apache. org*, 2011.

[31] Padro, Lluis, Stanilovsky, and Evgeny. Freeling 3.0: Towards wider multilinguality. In *LREC2012*, 2012.

[32] J. Piskorski and R. Yangarber. Information extraction: Past, present and future. in multi-source, multilingual information extraction and summarization. In *Past, present and future. In Multi-source, Multilingual Information Extraction and Summarization*, 2013.

[33] Diana Santos. Caminhos percorridos no mapa da portuguesificação: A linguateca em perspectiva. *Linguamática*, 1(1):25–58, 2009.

[34] Schmid and Helmut. Treetagger| a language independent part-of-speech tagger. *Institut fur Maschinelle Sprachverarbeitung, Universität Stuttgart*, 43:28, 1995.

[35] Michael Schmitz, Robert Bart, Stephen Soderland, Oren Etzioni, et al. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 523–534. Association for Computational Linguistics, 2012.

[36] Dan Tufis. Using a large set of eagles-compliant morpho-syntactic descriptors as a tagset for probabilistic tagging. In *LREC*, 2000.