Mestrado em Informática e Sistemas
Ramo de Desenvolvimento de Software

Search Engine Architecture

Trabalho realizado para a obtenção do grau de Mestre em Informática e Sistemas - Desenvolvimento de Software

Autor
Ana Sofia Silva Alves

Orientação
Doutor João Costa
ISEC
Engenheiro Leandro Braguês
doDOC

Coimbra, Agosto 2020

Resumo

Os constantes avanços na tecnologia proporcionam soluções que facilitam o processo da escrita de documentação, no entanto, particularmente em áreas técnicas e científicas, como a medicina e a indústria farmacêutica, a revisão e a associação de conceitos e técnicas, continuam a requerer um esforço dos intervenientes no seu processo de escrita. A plataforma doDOC surge com o intuito de ajudar profissionais a pouparem esforços neste processo, investindo sempre na procura de inovações, com vista a proporcionar melhorias contínuas. A extração de informação adequada é fundamental para analisar e interpretar este volume de informação de modo a agilizar e colaborar na escrita destes documentos.

Este estudo propõe um sistema que permite a pesquisa de termos em documentos e, retorna a definição do termo pesquisado e as relações que possam estar associadas ao mesmo. Para o atingir, foi realizado um estudo de aplicações utilizadas para extração de informação no contexto de *clinical trials*. O segundo passo incidiu num outro estudo de ferramentas que permitissem fazer indexação de estruturas e *Full Text Search*. Posto isto, foi criado um dicionário interno com base num repositório de informação médica selecionado: o *Metathesarus*, proveniente do UMLS. Seguiu-se o treino de vários modelos *multi-task* CNN, com os termos do dicionário criado, e a sua avaliação. Neste ponto, foi possível percorrer o conteúdo dos documentos da plataforma doDOC e proceder à criação de uma estrutura que armazene termos, juntamente com a sua definição, relações e localização no documento. Esta estrutura será periodicamente atualizada e indexada, possibilitando, numa fase final, que sejam realizadas pesquisas pelo utilizador, com uma resposta rápida e completa. Para o efeito, foi desenvolvido um *endpoint* para retornar o resultado da pesquisa ao utilizador.

Palavras-Chave: Extração de Informação; Estrutura de Dados; Pesquisa; Índices; *Full Text Search*

Abstract

The constant technological advances provide solutions that help the process of writing documentation, however, in technical and scientific areas such as medicine and the pharmaceutical industry, the revision process as well as the concepts and techniques association continue to require an effort from those involved in the writing process. The doDOC platform is designed to help professionals in this process, always investing in innovations to provide continuous improvements. The extraction of adequate information is fundamental to analyse and interpret this volume of information to speed up and collaborate in the writing of these documents.

This study proposes a system whose objective is to allow the search of terms in documents and to return the definition and the relationships that may be associated with the term searched. To achieve this, it surveys applications used to extract information in the context of clinical trials. Afterwards it focused on the evaluation of tools that would allow indexing structures and Full-Text Search. That said, an internal dictionary was created based on a selected repository of medical information: Metathesarus, from UMLS. This was followed by the training of CNN multi-task models, with the dictionary terms created, and their evaluation. At this point, it was possible to go through the content of the documents on doDOC platform and create a structure that would store terms along with their definition, relations and its position in the document. This structure will be systematically updated and indexed, enabling in a final stage, searches to be carried out by the user with a fast and complete response. For this purpose, an endpoint was developed to return the search results to the user.

Keywords: Information Extraction; Data structure; Research; Indexes; Full Text Search

Agradecimentos

A realização desta dissertação de mestrado contou com apoios fundamentais e aos quais estarei eternamente grata.

Ao Professor Doutor João Costa, pelo enorme apoio, orientação, disponibilidade, por todos os conhecimentos que me transmitiu, pelas ideias que trocámos, e por toda a sua colaboração no desenvolver deste trabalho.

Ao Leandro Braguês, orientador por parte da empresa doDOC, pela sua participação, orientação e ajuda na resolução de problemas.

Aos restantes colaboradores da doDOC, que estiveram sempre disponíveis para alguma necessidade que surgisse.

À minha irmã, ao meu cunhado e aos meus amigos, que estiveram cá mesmo quando faltou vontade para continuar, em especial, à Cristina, ao Gabriel e ao Filipe.

Por último, tendo consciência de que não poderia ter alcançado nada disto sozinha, dirijo um agradecimento muito especial aos meus pais, pelo incentivo, pelo apoio, e por tudo o resto!

Índice

Resumo i

Abstract iii

Agradecimentos v

Lista de Figuras ix

Lista de Tabelas xi

Acrónimos xiii

1 Introdução 1

1.1. Motivação 1

1.2. Objetivo 2

1.3. Contribuições 3

1.4. Estrutura 4

2 A Plataforma doDOC e o Protótipo da Solução 5

2.1 A Plataforma doDOC 5

2.1.1 Editor 5

2.1.2 Explorador de Ficheiros 8

2.2 Protótipo / Caso de Uso de Utilização 9

3 Estado da Arte 11

3.1. Extração de Informação 11

3.2. Aplicações de IE no Contexto de *Clinical Trials* 14

3.2.1. Aplicações de IE 15

3.2.2 Ferramentas de NLP 19

3.3. Repositórios Externos de Informação 21

3.4. Full Text Search e Indexação de Dados 22

4 Proposta de Extração e Integração de Informação 27

4.1. Recolha/Integração de Repositórios Externos (1ª Fase) 28

O Repositório Metathesaurus 28

4.2. Integração no MongoDB (2ªfase) 32

4.3. Treino do Modelo (3ªfase) 35

4.4. Criação da Estrutura e Indexação (4ª Fase) 42

4.5. Desenvolvimento do *Endpoint* (5ª Fase) 49

5. Conclusões 53

Referências 56

Apêndice A – Descrição dos Atributos (Metathesaurus) 1

Apêndice B – JSON Schema (Metathesaurus Subset) 1

Apêndice C – JSON Schema (Estrutura Final) 1
Apêndice D – Information Extraction Applications for Clinical Trials: A Survey 3

Lista de Figuras

Figura 1 - Editor de Documentos da plataforma doDOC 6
Figura 2 - Galeria de Templates 7
Figura 3 - Explorador da plataforma doDOC 8
Figura 4 - Estrutura de elementos 8
Figura 5 - Mockup do Sistema de Pesquisa 9
Figura 6 - Ligação entre as áreas de *Text Minning*. Fonte: [1] 11
Figura 7 - Exemplo do uso de Extração de Informação. Fonte: [2] 13
Figura 8 - *Clinical Trials*: Fases. Fonte: [7] 14
Figura 9 - Arquitetura da Solução Proposta 27
Figura 10 - Fluxograma para a construção do conjunto de dados 33
Figura 11 - Mapeamento dos atributos para o dicionário interno 34
Figura 12 – Exemplo de Dados de Treino - spaCy 36
Figura 13 - Exemplo de Dados de Treino (melhorado) 38
Figura 14 - Comparação de F-score dos Modelos 40
Figura 15 - Comparação do Tempo de Execução dos Modelos 40
Figura 16 - Nº de Entidades Encontradas pelos Modelos 41
Figura 17 - Evolução do F-score do Modelo 8 41
Figura 18 - Estrutura dos documentos da plataforma 43
Figura 19 – Identificação de relações no spaCy 45
Figura 20 - *Dependency Parser* – Abordagem 1- *Verb Approach* 46
Figura 21 - *Dependency Parser* - Abordagem 2 - *Aux Approach* 46
Figura 22 - *Dependency Parser* - Abordagem 3 - *Noun Approach* 47
Figura 23 - Abordagem usada no Endpoint 50
Figura 24 - Demonstração da Resposta do Endpoint através do Postman 51
Figura 25 - *Mockup* do Protótipo 52

Lista de Tabelas

Tabela 1 - Aplicações de Extração de Informação 18
Tabela 2 - Avaliação de métricas quantitativas 19
Tabela 3 - Tabela comparativa NLP. Adaptada de: [37] 20
Tabela 4 - Comparação de Ferramentas para *Full Text Search* 23
Tabela 5 - Exemplos de átomos no *Metathesaurus*. Adaptado de: [54] 29
Tabela 6 - Ficheiros *Metathesaurus*. Adaptado de: [54] 29
Tabela 7 - Listagem de ficheiros *Metathesaurus* [53] 30
Tabela 8 - Avaliação dos modelos de treino – spaCy 39
Tabela 9 - Definições do spaCy 44

Acrónimos

AMA - American Medical Association

CMS - Centers for Medicare and Medicaid Services

CNN – Convolutional Neural Network

CPT - Current Procedural Terminology

CPMC - California Pacific Medical Center

DFS - Depth First Search

EHR – Electronic Health Record

HITExt - Healthy Information Text Extraction

ICD10CM - International Classification of Diseases, Tenth Revision, Clinical Modification

IE – Information Extraction

KMCI - KnowlegdeMap Concept Indexer

LOINC - Logical Observation Identifiers Names and Codes

MeSH - Medical Subject Headings

NCHS - National Center for Health Statistics

NLM – U.S National Library of Medicine

OBIE – Ontology-Based Information Extraction

ODBC – Open Database Connectivity

OpenIE – Open Information Extraction

UIMA - Apache Unstructured Information Management Architecture

UMLS – Unified Medical Language System

1 Introdução

Atualmente, devido à globalização e ao advento de uma variedade de aplicações e soluções tecnológicas que permitem a escrita de documentos, o processo de escrita, verificação e validação colaborativa é imensamente simplificada. Estas soluções facilitam o processo, no entanto, particularmente em áreas técnicas e científicas, como a medicina e a indústria farmacêutica, a revisão e a associação de conceitos e técnicas, continuam a requerer um esforço dos intervenientes na escrita da documentação. O volume de documentação atualmente produzida, a contínua evolução e a descoberta de novas substâncias, métodos, aplicações e estudos, introduzem sérios obstáculos à produtividade na elaboração desses documentos. A extração de informação adequada é fundamental para analisar e interpretar este volume de informação de modo a agilizar e colaborar na escrita dos mesmos.

Esta dissertação tem com objetivo analisar potenciais abordagens e propor uma estratégia de extração de informação que seja integrável com uma plataforma de escrita colaborativa, e assim permitir agilizar tanto o processo de escrita como o de verificação e validação de documentos.

As secções seguintes apresentam a motivação, o objetivo, as contribuições desta dissertação e a estrutura deste documento.

1.1. Motivação

Soluções de escrita colaborativa de documentos, particularmente de documentos técnicos, permitem facilitar e agilizar a produção dos mesmos, particularmente em áreas de atividade técnicas e científicas, tais como a medicina e a indústria farmacêutica. A plataforma doDOC é um exemplo dessas soluções.

A plataforma doDOC é uma solução desenhada pela empresa homónima, doDOC, sediada em Boston, nasceu em 2015 e oferece, não só, mas principalmente a empresas farmacêuticas uma tecnologia que visa otimizar processos documentais. A missão da empresa é agilizar o processo colaborativo para a criação e revisão de documentos, eliminando tarefas que prejudicam a produtividade dos criadores de documentos.

Ao longo do documento, as referências da doDOC no restante documento referem-se à plataforma.

A doDOC é uma plataforma destinada à escrita colaborativa de documentos complexos. Consideram-se documentos complexos aqueles que envolvem a colaboração de um grande número de pessoas e, que exigem um maior esforço respetivamente à formatação. São normalmente documentos que requerem a revisão por vários intervenientes e cujo conteúdo é sensível e não pode ter falhas. Todo o conteúdo, que pode ser elaborado colaborativamente,

incluindo texto, figuras, tabelas, entre outros, sendo gerido e guardado pela plataforma, de modo a permitir que os utilizadores se possam dedicar e focar no conteúdo da informação. Esta plataforma também funciona como um repositório desses documentos colaborativos e, adicionalmente, de outros tipos de ficheiros, como por exemplo, pdfs, imagens, docx e vídeos.

A aplicabilidade desta plataforma abrange diferentes áreas, no entanto, neste momento o *target* de clientes está focado na indústria farmacêutica. Devido à quantidade de documentação produzida nesta área, colaborativamente por diversos utilizadores, sentiu-se a necessidade de criar um sistema que permita a pesquisa de termos dentro destes documentos. O capítulo 2 descreve mais detalhadamente a plataforma doDOC.

Esta dissertação enquadra-se no âmbito do Mestrado de Informática de Sistemas, ramo de Desenvolvimento de Software do Instituto Superior de Engenharia de Coimbra, sob orientação do Professor Doutor João Costa.

1.2. Objetivo

O objetivo deste trabalho é propor e implementar um sistema de pesquisa para a plataforma doDOC que permita facilitar o processo de elaboração do conteúdo dos documentos, a correlação de informação e a referenciação com outros documentos relacionados. Para isso, o sistema deve ter em consideração não só o repositório dos documentos registados e elaborados na plataforma, mas também incorporar informação de fontes de dados externas que sejam relevantes para o domínio da documentação que esteja a ser produzida. Isto permitirá que a pesquisa retorne informação técnica relevante, por exemplo, a definição do termo de pesquisa (se existir uma fonte em que esta esteja disponível), assim como algumas sugestões relacionadas com esse termo, e ainda os documentos onde o mesmo se encontra.

Por exemplo, supondo que existe o Medicamento A e a Substância B. A Substância B aparece várias vezes nos documentos da plataforma como sendo uma substância ativa do Medicamento A. Quando o utilizador introduz como termo de pesquisa “Medicamento A”, o resultado que o mesmo obterá será a listagem dos documentos onde “Medicamento A” está referenciado, a definição desse medicamento e ainda a informação de que o Medicamento A está relacionado com a Substância B, apresentando ao utilizador a opção de fazer uma pesquisa sobre a Substância B.

Para alcançar esse objetivo, inicialmente será elaborado um estudo relativamente a tecnologias usadas para a extração de dados (*Information Extraction* e *Relation Extraction*).

Apesar de não ser o único foco da plataforma doDOC, neste momento, esta encontra-se a ser utilizada por grandes empresas farmacêuticas que produzem uma grande quantidade de documentação, entre relatórios e artigos, destacando-se o uso da plataforma para a escrita de *clinical trials*.

Por esse motivo, e porque se pretende demonstrar a aplicabilidade da solução proposta na forma de protótipo, é avaliada a aplicabilidade dessas tecnologias de extração de informação no contexto de *clinical trials*.

Serão também tomados em consideração os repositórios de informação externos considerados relevantes para este contexto, que incluam entidades e potenciais relacionamentos, nomeadamente do UMLS (*Unified Medical Language System).* Este é um sistema que produz e distribui fontes de conhecimento médicas (bases de dados). Estes repositórios são multiuso, isto é, não são otimizados para uma aplicação em específico e podem ser aplicados em vários sistemas.

De seguida será proposto um processo de extração dos dados dos documentos e uma estrutura de dados que irá permitir armazenar informações adicionais sobre as entidades encontradas, como por exemplo, a sua definição e semântica. Posteriormente, será feito um estudo para perceber qual a melhor ferramenta que permita indexar documentos, as estruturas de dados extraídas e permitir efetuar pesquisas com *Full Text Search* sobre os dados indexados. *Full Text Search* é uma técnica que usa índices de texto para efetuar a pesquisa, tornando-a mais rápida.

Com este sistema, o processo de construção/edição de documentos será mais produtivo, pois permite adicionar um conjunto de funcionalidades que, até este momento, não possuía. Exemplo disso é a pesquisa, dentro da plataforma doDOC, da definição de um termo, que atualmente o utilizador necessita de recorrer a ferramentas externas para obter essa informação. Além da definição de um termo pesquisado, permite também indicar o local onde este foi referenciado e onde pode ser encontrado.

1.3. Contribuições

Este trabalho teve como resultado as seguintes contribuições:

- Análise de ferramentas de *Information Extraction* (IE), particularmente com *clinical trials*, a sua adequação no contexto da solução proposta culminando com a publicação e apresentação oral do artigo "*Information Extraction Applications for Clinical Trials: A Survey"* publicado em CISTI 2019 – 14ª Conferência Ibérica de Sistemas e Tecnologias de Informação e disponível na Biblioteca Digital IEEE Xplore, ISI, Scopus, EI-Compendex, INSPEC e Google Scholar;
- Estudo de possíveis repositórios de informações externos para a construção de um dicionário interno com termos médicos;
- Análise do modo de funcionamento da fonte externa *Metathesaurus* e proposta e construção de um dicionário com termos médicos;
- Proposta de um modelo de extração de informação, que inclui três abordagens para extrair informação com o spaCy;

- Treino de vários modelos *multi-task CNN* do spaCy com entidades recolhidas do dicionário interno. É apresentada uma análise de resultados da avaliação do treino desses modelos;
- Análise de ferramentas para *Full Text Search* e indexação de dados;
- Integração em ambiente de produção na plataforma a funcionalidade de *Full Text Search*;
- Proposta e construção de uma estrutura genérica para indexação;
- Definição e desenvolvimento de um *endpoint* que, quando recebe uma *query* do utilizador, pesquisa na estrutura indexada e retorna os resultados.

1.4. Estrutura

O presente relatório divide-se em cinco capítulos: Introdução, A Plataforma doDOC e o Protótipo da Solução, Estado da Arte, Proposta de Extração e Integração de Informação e Conclusões.

O segundo capítulo descreve a plataforma doDOC, onde se irá integrar a solução proposta, e é apresentado o protótipo do sistema e as suas funcionalidades.

O terceiro capítulo apresenta um estado da arte sobre aplicações e ferramentas para extração de informação, com especial foco em Processamento de Linguagem Natural, e ferramentas de Indexação.

O quarto capítulo descreve a solução proposta que é composta por 4 fases: a recolha de informação de dados de um repositório de informação externo, com o propósito de criar um dicionário interno à plataforma; a sua integração num repositório; o treino de vários modelos com dados provenientes de outros documentos relevantes para o contexto e do dicionário interno de termos; a criação de uma estrutura para guardar a informação extraída do modelo escolhido; e, por último, a disponibilização da informação, através de um *endpoint*, para as pesquisas do utilizador.

Finalmente, no quinto e último capítulo, são apresentadas algumas reflexões sobre as contribuições da solução proposta, as contribuições da tese, as limitações encontradas e o trabalho futuro.

2 A Plataforma doDOC e o Protótipo da Solução

Este capítulo apresenta a plataforma doDOC, algumas das suas funcionalidades principais e a sua organização, assim como o protótipo com as funcionalidades pretendidas.

2.1 A Plataforma doDOC

A plataforma doDOC é destinada à escrita colaborativa de documentos complexos. Documentos complexos[1] são documentos em que uma ou mais das seguintes caraterísticas se aplica:

- Complexidade no conteúdo: o documento é extenso e/ou contém muitas imagens, equações, tabelas ou citações;
- Complexidade no *workflow*: é necessário que muitas pessoas colaborem no documento, seja sequencialmente ou simultaneamente, com diferentes permissões durante o ciclo de vida do documento;
- Complexidade no *design*: o documento tem definições restritas de *design*, incluindo fontes de letra, margens, cabeçalhos/rodapés, entre outros, e necessitam de cumprir com requisitos internos e externos.

A plataforma disponibiliza um editor colaborativo de documentos (secção 2.1.1), e um repositório de pastas, documentos e outros ficheiros (secção 2.1.2).

2.2 Editor

A interface da plataforma doDOC disponibiliza um editor WYSIWYG (*What You See Is What You Get*), colaborativo, onde os vários utilizadores podem, simultaneamente, trabalhar no mesmo documento.

Para a edição de documentos, o editor apresenta várias funcionalidades, incluindo a inserção de notas, comentários, índices, referências cruzadas e citações. Paralelamente, é possível usar

[1] Definição da empresa doDOC

funcionalidades como verificação linguística e a atribuição de permissões ao nível de parágrafos. Desta forma, é possível gerir e filtrar, dentro dum documento, quais os utilizadores que podem visualizar, editar ou comentar partes do conteúdo de um documento. É de realçar a funcionalidade *track changes*, que permite aos utilizadores manterem um registo das alterações que foram efetuadas. Todas as funcionalidades mencionadas acima podem ser usadas colaborativamente.

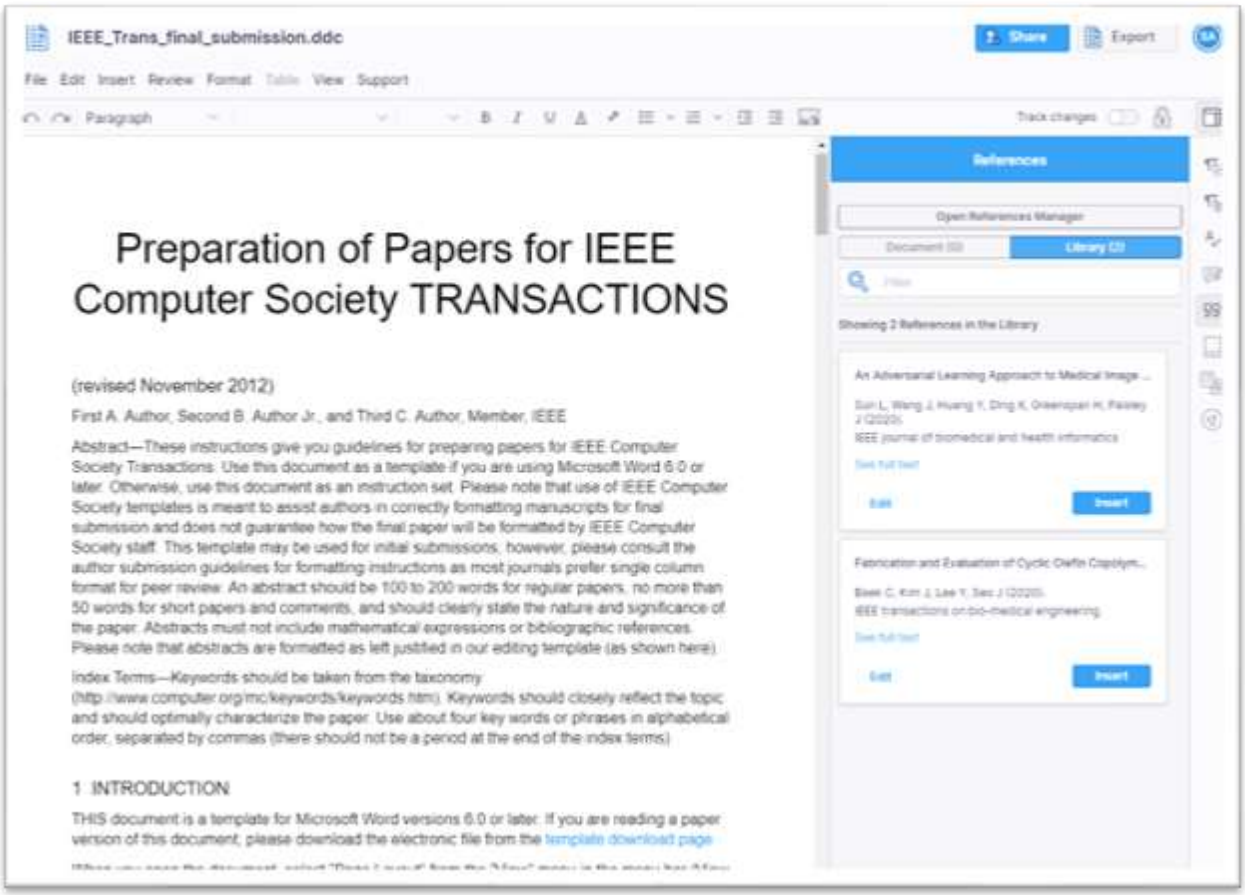

Figura 1 - Editor de Documentos da plataforma doDOC

A Figura 1 ilustra o editor disponibilizado pela plataforma. No menu, que se encontra acima do texto, são visíveis os ícones e as opções de formatação e, à direita, encontra-se uma secção com a lista de citações que estão associadas ao documento.

Uma das maiores vantagens da plataforma é permitir ao utilizador focar-se no conteúdo que está a escrever/editar, pois toda a informação que é introduzida no documento pode ser exportada para diferentes formatos e estruturas de acordo com o *template* escolhido. Esta vertente é uma característica distintiva e de extrema relevância, particularmente no contexto da indústria farmacêutica, pois sempre que um novo medicamento é criado, é necessário submeter pedidos de aprovação desse medicamento a diversas entidades, normalmente com formatos distintos. Além disso, esta aprovação tem um carácter nacional, sendo necessário repetir este processo para as várias entidades em cada um dos países em que se pretende comercializar esse

medicamento. A plataforma, ao permitir guardar toda a estrutura de informação e formatação, possibilita que, através da utilização de uma lógica de *templates*, este processo de criação da documentação necessária, seja amplamente simplificado, pois basta exportar de acordo com o *template* específico de cada uma destas entidades. A Figura 2 ilustra a galeria de *templates* que o utilizador tem acesso quando exporta o seu documento.

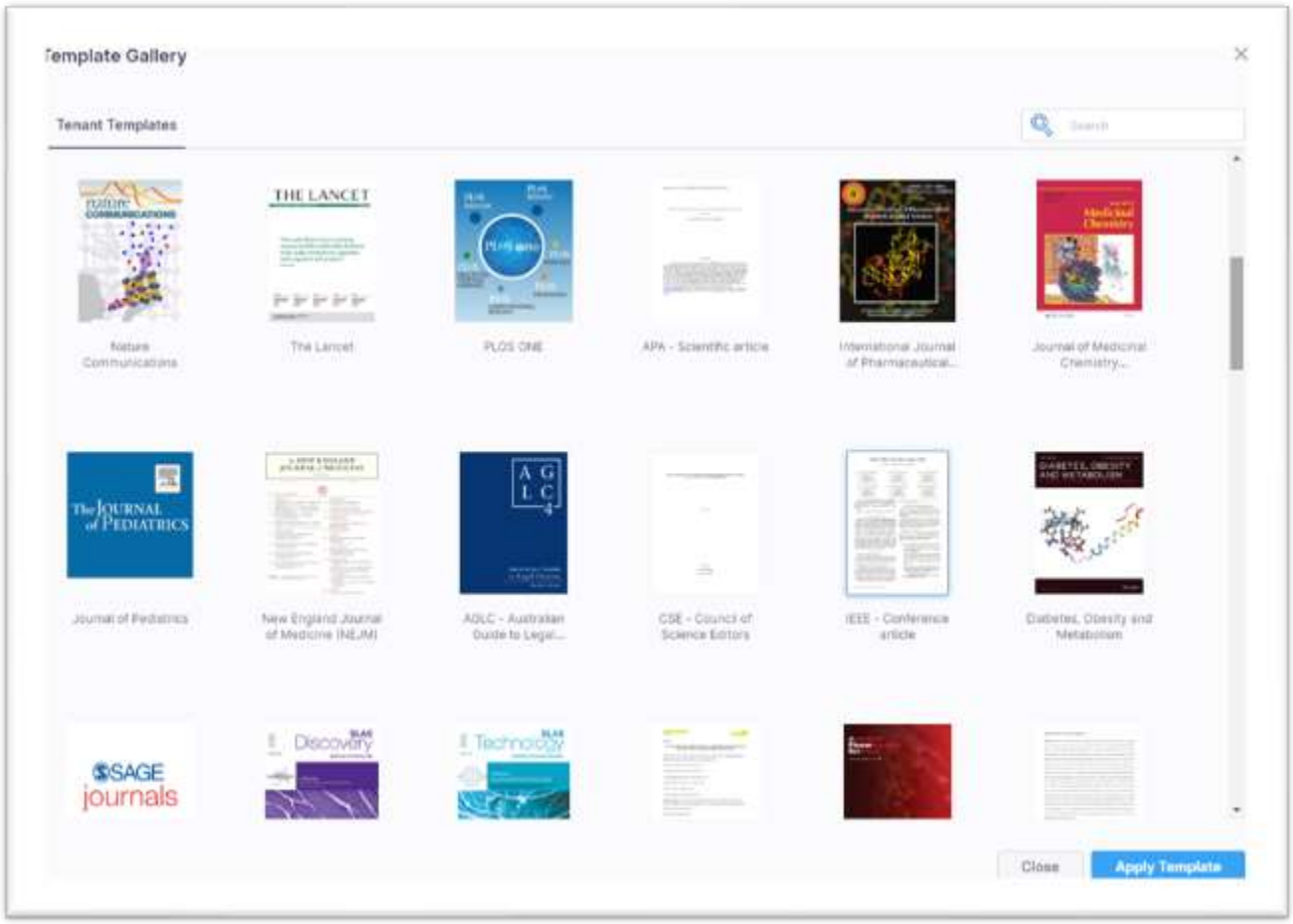

Figura 2 - Galeria de Templates

2.1.2 Explorador de Ficheiros

A plataforma organiza os documentos numa estrutura composta por espaços (*spaces*), pastas (*folders*), documentos (*documents*) e ficheiros (*files*). A Figura 3 ilustra o gestor que permite visualizar e proceder a essa organização.

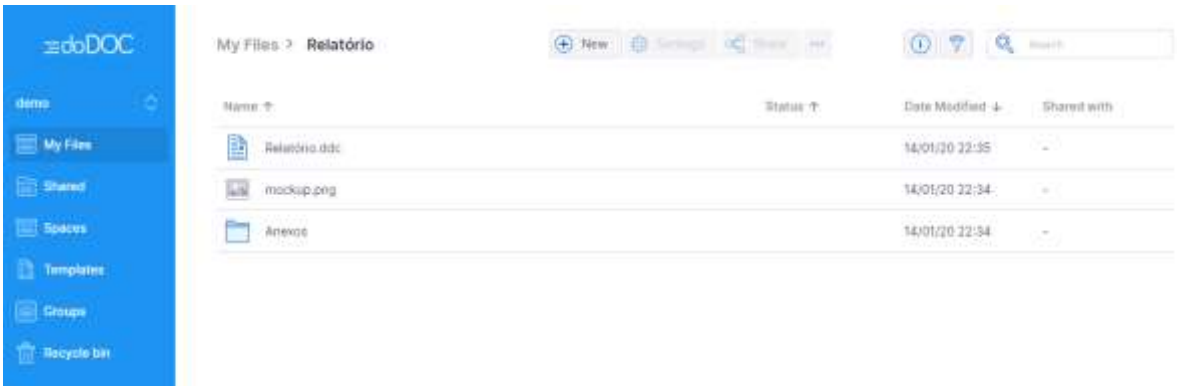

Figura 3 - Explorador da plataforma doDOC

Estes elementos representam uma hierarquia, como é ilustrado na Figura 4. Sendo que a raiz da estrutura são os *spaces*. Existe sempre o *space* pessoal do utilizador, todavia podem ser criados vários, a fim de serem partilhados com outros utilizadores. Dentro de um *space* pode existir qualquer elemento, isto é, *folders*, *documents* e *files*. Os *documents* e *files* podem ser também guardados dentro das *folders*.

No caso dos documentos, há três formas possíveis para proceder à sua criação. Pode ser criado um documento vazio, a partir de um *template* ou ser importado do formato docx.

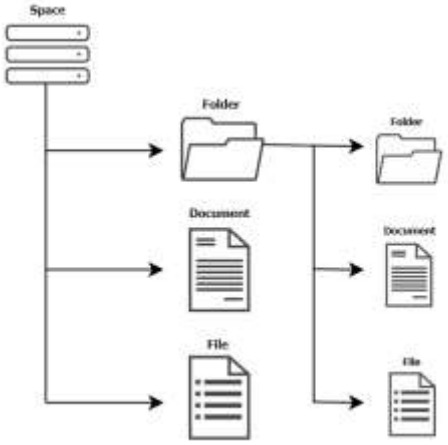

Figura 4 - Estrutura de elementos

Para cada um dos elementos, é possível fazer a partilha com outros utilizadores bem como atribuir permissões. Há vários níveis de permissões possíveis, por exemplo, o utilizador cria uma *folder* e torna-se automaticamente *owner* dela. Após isso, o utilizador partilha a pasta com outro, e pode dar-lhe permissão de acesso, de edição ou de *owner* também.

2.2 Protótipo / Caso de Uso de Utilização

A solução proposta deve possibilitar ao utilizador a disponibilização de informação relacionada com determinados termos de pesquisa, que sejam relevantes no contexto do documento em que esta pesquisa é efetuada. A informação disponibilizada deve tomar em consideração não só a informação proveniente de documentos geridos e mantidos internamente na plataforma doDOC, mas também proveniente de repositórios de informação que sejam relevantes para o contexto do documento onde a pesquisa está a ser efetuada. A Figura 5 apresenta um *mockup* da funcionalidade, que ilustra, sem grande detalhe, a funcionalidade que se pretende oferecer e integrar na plataforma doDOC. À direita é possível visualizar uma área especifica de navegação onde o utilizador deve digitar o termo que pretende pesquisar.

Após efetuar a pesquisa, o utilizador deve obter, sempre que possível, a definição do termo, bem como todos os documentos onde seja feita referência a esse termo, incluindo o documento que está a ser editado. Deste modo, o utilizador poderá posteriormente pesquisar entre os termos que estejam de alguma forma relacionados com o termo inicial da pesquisa.

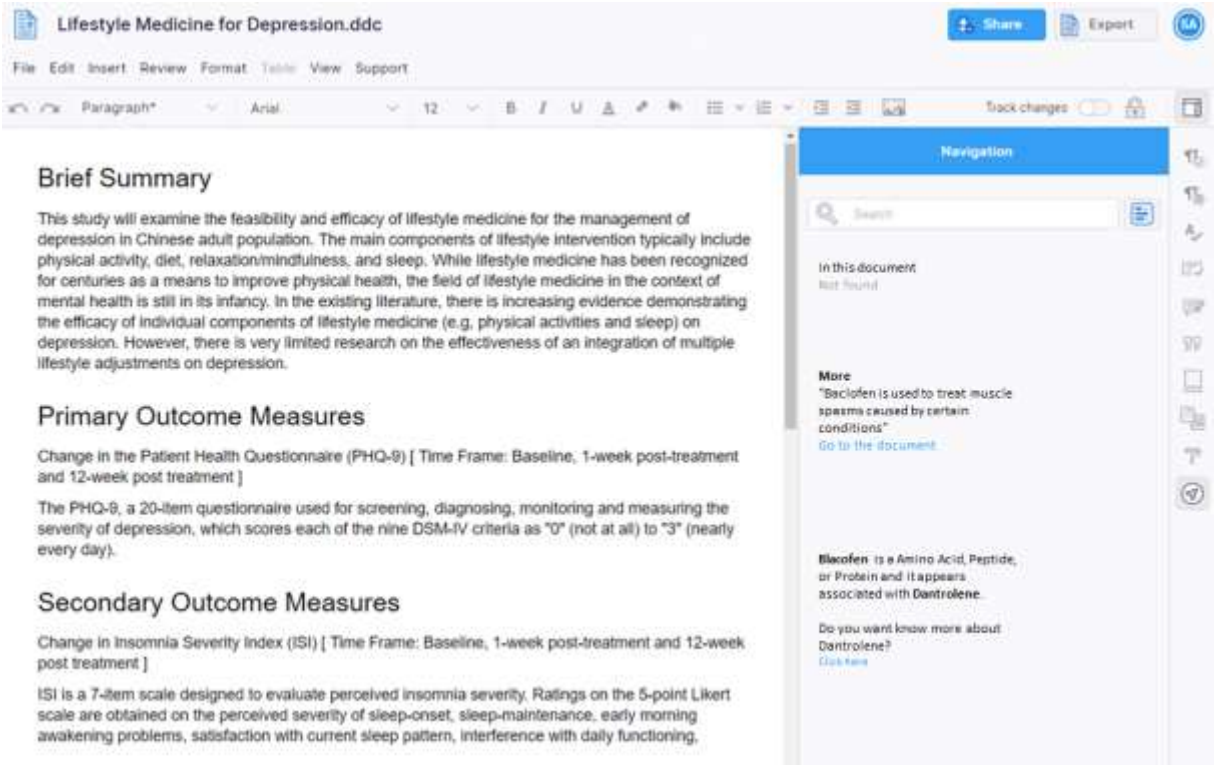

Figura 5 - Mockup do Sistema de Pesquisa

No exemplo ilustrado na Figura 5, quando o utilizador pesquisar por "*Baclofen*", este obtém a informação de que o termo pesquisado se trata de um aminoácido, péptido ou proteína. Para além da definição, é apresentada a informação de que não há referências desse termo no documento que está a ser editado, no entanto, uma vez que o termo existe noutro documento, é-lhe fornecido a opção de navegar para onde o termo se encontra. Por último, é também

apresentada informação de que o termo de pesquisa, "*Baclofen*", aparece frequentemente associado ao termo "*Dantrolene*", juntamente com uma opção de pesquisa para esse termo.

A ferramenta integrada e/ou a desenvolver de tomar em consideração os seguintes requisitos:

- A ferramenta terá de ser integrável com a plataforma doDOC;
- Existe a necessidade de que os documentos internos à plataforma possam ser processados pela ferramenta;
- Deverá permitir a integração de dicionários externos;
- Preferencialmente deverá ser em Python, de fácil instalação e com uma curva de aprendizagem baixa;
- Aplicável noutros contextos;
- Suporte disponível.

Deste modo, é necessário identificar as ferramentas/soluções existentes e avaliar a sua aplicabilidade no contexto da plataforma e se necessário propor e implementar uma solução.

3 Estado da Arte

Este capítulo faz o enquadramento da extração de informação e análise das ferramentas existentes para extração de informação relacionadas com termos médicos que pudessem ser aplicadas no contexto de *clinical trials*. Apresenta um levantamento das fontes de informação externa que podem ser integradas e relevantes para o contexto de *clinical trials*. Por fim, é feita uma análise sobre aplicações para *Full Text Search*.

3.1. Extração de Informação

Extração de Informação é uma das áreas do *Text Mining*. *Text Mining* refere-se ao processo de extração de informação de texto e, com base em características, pode ser dividida em sete áreas diferentes. Sendo estas:

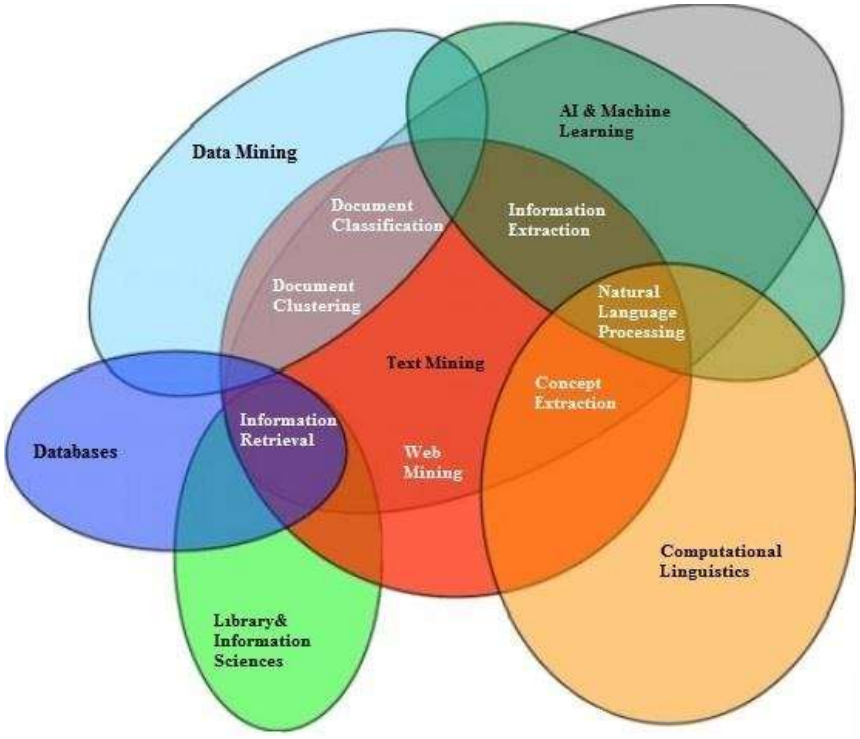

Figura 6 - Ligação entre as áreas de *Text Minning*. Fonte: [1]

Search and Information Retrieval centram-se no armazenamento e recuperação de texto de bases de dados;

Document Clustering agrupa e categoriza termos, parágrafos ou documentos usando métodos de *data mining*;

Document Classification agrupa fragmentos, parágrafos ou documentos, usando *data mining*;

Web Minning envolve *data mining* e *text mining* de informação retirada da Internet, tendo como foco a escalabilidade e interconetividade proveniente da web[2].

Natural Language Processing foca-se em linguagem de baixo nível para processar e perceber *tasks* com o intuito de converter ocorrências de linguagem humana em representações formais, manipuláveis por computadores. Isto é, NLP é um componente de *Text Mining* que, através de várias análises, ajuda a máquina a "ler" e processar texto escrito por um humano;

Concept Extraction permite a criação de grupos de palavras e frases em grupos semanticamente semelhantes através da extração e da sua definição;

Information Extraction contemporiza a identificação e extração de factos revelantes e relações de texto não-estruturado, como de crónicas ou notícias [1].

A Figura 6, extraída de [1], ilustra a ligação entre as áreas acima descritas.

As aplicações de *Information Extraction* (IE) geralmente envolvem um conjunto de tarefas sequenciais:

- O texto deve ser pré-processado, onde é preparado para processamento com a ajuda de algumas ferramentas como *tokenization*, divisão de frases, análise morfológica, etc.;
- Os conceitos são extraídos e classificados - por exemplo, menções de pessoas, locais, eventos entre outros;
- Os conceitos relacionados são vinculados quando um relacionamento é identificado entre eles;
- Posteriormente, os dados duplicados são removidos e finalmente, o conhecimento extraído é normalmente inserido numa base de dados para uso posterior.

A extração de informação pode ser totalmente automatizada ou realizada com a ajuda de *input* humano. Para uma melhor perceção desta área, a Figura 7, extraída de [2], ilustra um exemplo de como a extração de informação funciona.

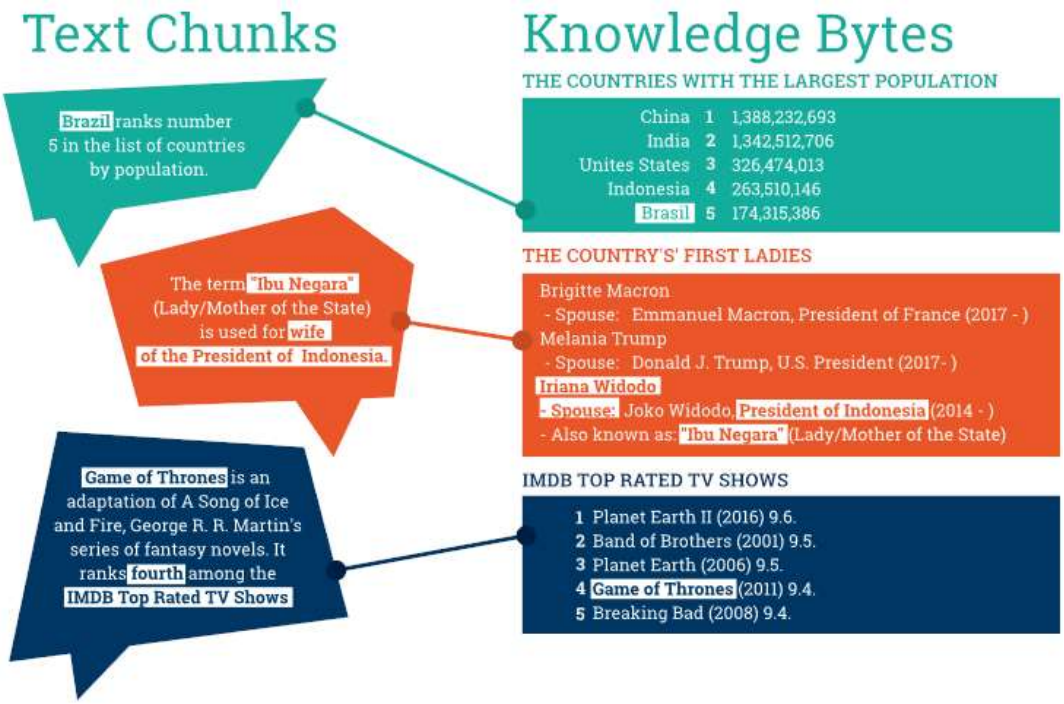

Figura 7 - Exemplo do uso de Extração de Informação. Fonte: [2]

Dentro do IE, existem abordagens diferentes sendo que as mais relevantes estão definidas abaixo:

- ***Unsupervised learning system*** é uma abordagem usada para métodos que normalmente dependem de pequenas regras para aprenderem. É um modelo que usa apenas dados de entrada e nenhuma variável de saída. Normalmente é usado para problemas de *clustering* e de *association* [3];
- ***Supervised learning system*** é um método que requer grandes quantidades de dados para treino. Neste método existem várias variáveis de entrada e uma de saída e, o objetivo é a inferência de uma função que faça o mapeamento entre esses dados [3];
- ***Rule-based systems*** são uma categoria de *OpenIE*. *OpenIE* é uma variante de IE, que difere principalmente na forma como as relações são extraídas. Isto é, o esquema para essas relações não precisa de ser especificado com antecedência; normalmente o nome da relação é o texto que liga dois argumentos. Este método extrai todos os tipos de relações encontradas num texto. Para isso, estes sistemas fazem uso de regras de extração manuais[4].
- ***Clause-Based Systems*** são também uma categoria de *OpenIE*. Visam melhorar a precisão e baseiam-se na ideia de reestruturar uma frase. O objetivo é transformar frases complexas com relações em num conjunto de cláusulas independentes sintaticamente simplificadas e fáceis de interpretar[4].

- ***Convolutional Neural Network*** (CNN) é um método semelhante às *Neural Networks* tradicionais, isto é, funciona através de uma sequência de camadas e cada uma delas transforma um dado de entrada, num de saída com uma função diferenciável que pode ou não ter parâmetros. É normalmente usada para identificar imagens, no entanto, quando aplicada a NLP também obtém bons resultados[5].
- ***Ontology-Based Information Extraction*** é um dos subcampos da extração de informação. Neste método, as ontologias – que fornecem uma especificação formal e explícita de um conceito – desempenham um papel crucial no processo de IE, uma vez que são usadas pelo processo de extração. Como a extração de informação está relacionada com a tarefa de recuperar informações para um certo domínio, a especificão formal e explicita dos conceitos desse domínio através de uma ontologia é útil para esse processo [6].

3.2. Aplicações de IE no Contexto de *Clinical Trials*

Esta secção pretende apresentar um estudo sobre aplicações de IE que poderiam ser utilizadas no contexto de *Clinical Trial*s.

Os *Clinical Trials*, ou ensaios clínicos, são estudos que avaliam a eficácia de novos medicamentos experimentais, dispositivos médicos ou procedimentos feitos em determinados grupos controlados de pessoas. Como ilustrado na Figura 8, extraída de [7], estes estudos dividem-se em fases distintas, com um número crescente de participantes e, envolvem um grande volume de informação.

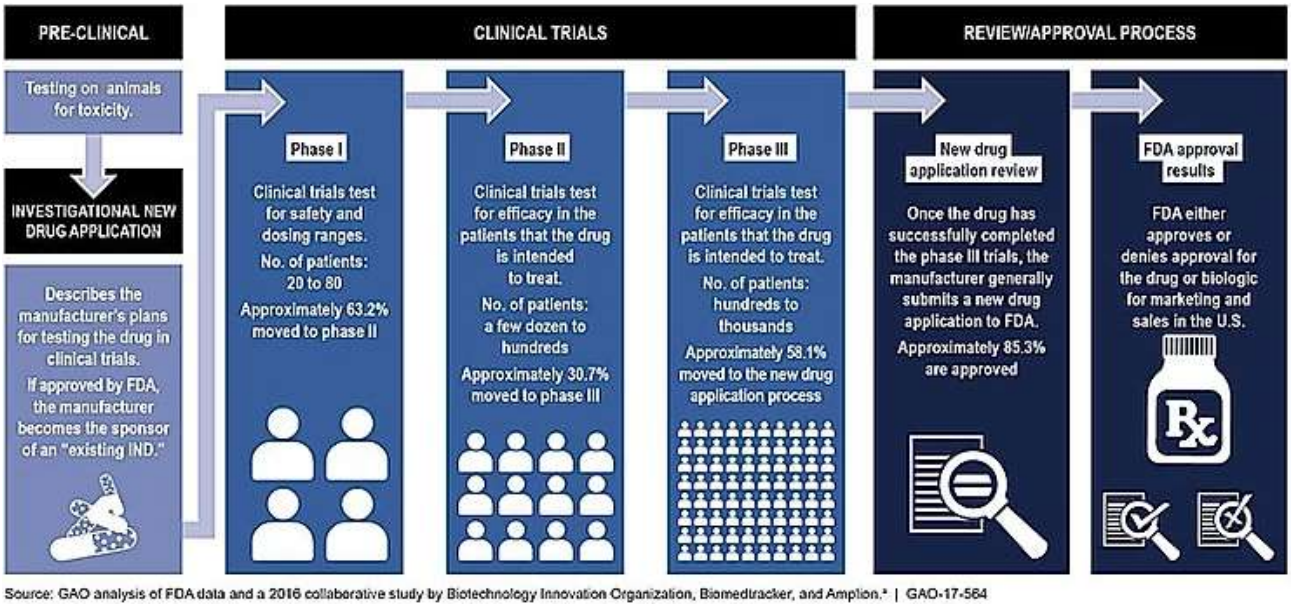

Figura 8 - *Clinical Trials*: Fases. Fonte: [7]

3.2.1. Aplicações de IE

Relativamente às aplicações utilizadas neste contexto, a metodologia usada foi identificar, através de um estudo bibliográfico, as aplicações mais utilizadas e com melhores referências. Este estudo foi feito, independentemente dos requisitos propostos e apresentados (secção 2). Por conseguinte, algumas aplicações foram avaliadas, tendo sido instaladas e exploradas.

Quanto às aplicações estudadas, segundo a abordagem de *Open* IE, mais especificamente o método *Learning-based System, Mausam* [4][8] apresentou OLLIE (*Open Language Learning Information Extraction*), o primeiro *Open* IE a identificar não apenas relações baseadas em verbos, como também relações de substantivos e adjetivos. *TextRunner* [9] e WOE [10] são aplicações desenvolvidas que seguem a mesma abordagem.

ReVerb [11] e, mais recentemente, KrakeN [4] [12], Exemplar [13] e PredPatt [14] são exemplos de *Rule-based Systems*. KrakeN foi a primeira abordagem para criar factos a partir de frases. Esse acontecimento deve-se ao facto de reunir um conjunto completo de argumentos para cada relação numa frase.

PredPatt aplica um conjunto de regras não-lexicais definidas em *Universal Dependencies* [15]. Esse conjunto de regras é usado para extrair estruturas de predicado-argumento. Dessa forma, um gráfico é construído entre o *token* do predicado e os *tokens* dos argumentos.

ClauseIE [4][16] é um *Clause-based system* que explora o conhecimento linguístico presente na gramática da língua inglesa. Isso permite mapear as relações de dependência numa frase para cláusulas.

Ainda no contexto de aplicações Open IE, Niklaus et al [4] analisaram NestIE [17], MinIE [18] e Graphene [19].

Gutierrez [20] apresenta duas extensões para o método OBIE. A metodologia usa ontologias para orientar o processo de extração de informações. Essas duas extensões levam a uma abordagem híbrida para o processo de extração. A primeira aumenta a precisão do processo OBIE através da combinação de extratores de informações com diferentes estratégias de implementação. A segunda extensão adiciona uma funcionalidade para identificar elementos incorretos no texto.

Li [19] anuncia uma implementação de uma ferramenta de IE para informação clínica com o método CNN. Essa implementação usa uma *deep neural network* para anotar períodos de eventos e seus atributos a partir de notas clínicas brutas e relatórios de patologias.

Wang et al [21] exploraram ferramentas para extração de informações em informações clínicas. De acordo com este artigo, as ferramentas mais utilizadas para o IE nos domínios clínicos são cTakes [22], MetaMap [23] e MedLEE [19][24], ordenadas por frequência de uso. Outras ferramentas identificadas são KMCI (*KnowlegdeMap Concept Indexer*) [25], HITEx [26], MedEx [27], Medtex, MedXN, MedTime e as *frameworks* UIMA, GATE, Protégé e MedTagger.

Começando com a ferramenta cTakes [21], desenvolvida pela Mayo Clinic e a mais utilizada dentro deste contexto. Foi desenvolvida com base em vários projetos *open source* e na *framework* UIMA[22] e ainda sobre o Apache OpenNLP [24]. Contém mecanismos de análise para várias tarefas clínicas e linguísticas, como deteção de frases, *tokenization*, tags de *part-of-speech*, deteção de conceitos e normalização dos mesmos. Esta ferramenta foi adotada para identificação de coortes fenotípicos de pacientes, extração de estado de tabagismo, estudos de associação em genomas, extração de contraindicações de medicamentos, deteção de discrepâncias de medicamentos, descoberta de relações temporais e identificação de fator de risco de EHRs.

MetaMap[23] é uma ferramenta desenvolvida pela *National Library of Medicine*. O objetivo do *MetaMap* é mapear texto biomédico para o UMLS *Metathesaurus* ou vice-versa. Primeiramente foi desenvolvido para melhorar a recuperação de texto biomédico das citações MEDLINE e *PubMed*. Mais tarde, foi aprimorado para processar texto clínico[28].

MedLEE [19][24] é a terceira ferramenta mais usada e é um dos primeiros sistemas clínicos de NLP usado principalmente para farmacovigilância e farmacoepidemiologia. Este sistema extrai, estrutura e codifica informações clínicas de relatórios de pacientes [29]. Atualmente, o MedLEE é usado diariamente para processar relatórios radiológicos de pacientes no CPMC.

KnowlegdeMap Concept Indexer [25] é um sistema de NLP que identifica e indexa conceitos e biomédicos e depois mapeia para conceitos UMLS. O KMCI processa documentos em três fases:

1. Identificação e normalização de frases;
2. Técnicas de identificação de conceitos e NLP;
3. Desambiguação de conceitos.

Esta ferramenta foi desenvolvida para extrair conceitos representados em textos educacionais médicos [25].

O HITExt [26] é um *software open source* de NLP desenvolvido por um grupo de investigadores do *Brigham and Women's Hospital* e da *Harvard Medical School*. É uma aplicação orientada a dados e foi usada com sucesso para processar registos médicos de duas instituições diferentes.

MedEx [27] é uma aplicação que tem sido usada em vários estudos. É um *rule-based system* que extrai o nome, força, rota e frequência dos medicamentos.

Ainda sobre aplicações de NLP, MedTex extrai informações relevantes de texto narrativo para facilitar a equipa clínica no processo de tomada de decisão [21]. Foi desenvolvida em parceria com profissionais de saúde dos departamentos relacionados com cancro, radiologia e medicina de emergência.

Por último, no domínio das aplicações, MedXN – *Medication Extraction and Normalization* - usada para extrair informações abrangentes sobre medicamentos de narrativas clínicas e normalizá-las para a RxNorm[2] [30]. O método utilizado para esta aplicação permitiu a extração de informação de descrições de medicamentos em notas clínicas que foram decompostas em nome e atributos do medicamento, extraídos separadamente usando a pesquisa no dicionário RxNorm.

No contexto de *frameworks*, a primeira a ser analisada é a *framework* UIMA [22]. As aplicações que têm UIMA como base, são sistemas que analisam grandes volumes de informações não estruturadas e paralelamente consultam fontes de informações estruturadas para descobrir, organizar e apresentar conhecimentos relevantes para o utilizador final. A arquitetura UIMA define funções, interfaces e comunicações de componentes de alta granularidade, essenciais para aplicações fazerem uso da mesma. Sucintamente, a *framework* permite que as aplicações sejam decompostas em componentes, por exemplo:

1. Identificação do idioma;
2. Segmentação específica do idioma;
3. Deteção de frases;
4. Deteção de entidades (nomes de pessoas, lugares, etc.).

GATE [31] é uma *framework* e um ambiente de desenvolvimento gráfico que permite aos utilizadores desenvolver componentes e recursos de forma robusta. Isto permite não apenas desenvolver aplicações para tarefas de processamento, como *Information Extraction*, mas também criar e anotar avaliações das aplicações desenvolvidas.

A *framework* Protégé [32] tem sido frequentemente utilizada para a construção de ontologias. Os programadores que pretendem usar essa estrutura podem integrar o *output* do Protégé com *rule-based systems* para construir uma ampla variedade de sistemas inteligentes. Também inclui classificadores dedutivos para validar que os modelos são consistentes e inferir novas informações com base na análise de uma ontologia.

O MedTime é uma *framework* para extrair informações temporais de narrativas clínicas e normalizá-las para o padrão TIMEX3[3] [26][33]. É híbrido com procedimentos entrelaçados de *machine learning* e *rule-based* para extração de informações temporais. É dividido em seis etapas: pré-processamento, marcação temporal através do HeidelTime, marcação de frequência clínica, marcação de sequência, normalização temporal clínica e pós-processamento.

[2] RxNorm é o nome de uma terminologia especifica dos EUA e contém todos os medicamentos disponíveis nesse mesmo local.
[3] TIMEX3 é uma *tag* usada para marcar expressões temporais explícitas, como horas, datas e durações.

MedTagger [5][34], outra *framework* utilizada no contexto de IE, inclui três componentes principais: MedTagger para indexação baseada em dicionários, MedTaggerIE para extração de informações com base em padrões e MedTaggerML para reconhecimento de entidades através de *machine learning*. É um *pipeline* de NLP *open source* baseado na *framework* UIMA para indexação baseada em dicionários, extração de informações e reconhecimento de entidades a partir de texto clínico.

A Dentro das que estão apresentadas na tabela, e segundo o estudo as que se destacam são MetaMap, MedLEE e cTakes. No Apêndice D encontra-se uma tabela que faz uma análise comparativa das ferramentas que têm mais casos de uso.

Tabela 1 apresenta as ferramentas de IE consideradas mais relevantes segundo [21], tendo sido as que são mais frequentemente utilizadas. Dentro das que estão apresentadas na tabela, e segundo o estudo as que se destacam são MetaMap, MedLEE e cTakes. No Apêndice D encontra-se uma tabela que faz uma análise comparativa das ferramentas que têm mais casos de uso.

Tabela 1 - Aplicações de Extração de Informação

Aplicação	***Descrição***	***Dataset***	***Tarefas***
UIMA	*Framework* para analisar conteúdos não estruturados	Conteúdos não estruturados: texto, vídeo e áudio	Aplicações de pesquisa
MetaMap	Aplicação para extrair conceitos biomédicos e transformá-los em UMLS (*Unified Medical Language System*)	Narrativas Clínicas	Extração de fenótipos Extração de atributos relacionados ao paciente
MedLEE	Sistema para extrair, estrutura e codificar informação clínica	Notas de narrativas clínicas	Farmacovigilância Farmacoepidemiologia
MedEx	Sistema para reconhecer nomes de medicamentos, doses, procedimento e frequência	Registos clínicos não estruturados (texto livre)	Extração de menções do uso da aspirina e informação da dosagem Estudos fármaco-genéticos
cTakes	Sistema IE *Open-Source* com base na *framework* UIMA	EHRs	Identificação de patentes Extração de eventos adversos de medicamentos
MedTex	Sistema IE para suporte de decisão de equipas clínicas	Relatórios de narrativas clínicas	Suporte de decisão

Decidiu-se instalar as ferramentas mais destacadas em [21], cTakes, MetaMap e MedLEE, e fazer uma análise qualitativa de vários aspetos, nomeadamente:

- Facilidade de Instalação: Baixa, Alta;
- Facilidade de Utilização: Baixa, Alta;
- Facilidade de Aplicação Noutros Contextos: Baixa, Alta;
- Curva de Aprendizagem: Alta, Baixa.

Foi possível instalar o cTakes e o MetaMap, não tendo sido possível instalar o MedLEE uma vez que não foi encontrada nenhuma versão disponível online. Esta análise encontra-se na Tabela 2.

Tabela 2 - Avaliação de métricas quantitativas

Métricas/Aplicação	***cTakes***	***MetaMap***	***MedLEE***
Facilidade de Instalação	Alta	Baixa	NA
Facilidade de Utilização	Alta	Baixa	NA
Facilidade de Aplicação Noutros Contextos	Baixa	Alta	NA

As aplicações tinham uma certa dificuldade na instalação/execução, alguma falta de suporte infraestruturas complexas. Algumas, como o caso do cTakes precisavam de muito tempo para processamento. Para o contexto da plataforma doDOC e avaliando a facilidade de utilização não se adequavam em termos de execução. Dados esses motivos e à necessidade de adaptação à arquitetura da plataforma doDOC, optou-se por fazer um levantamento de ferramentas com menor complexidade, com uma curva de aprendizagem reduzida e de fácil integração. Para o efeito é necessário encontrar e avaliar aplicações de NLP que possam ser usadas.

3.2.2 Ferramentas de NLP

A Tabela 3 compara as seguintes ferramentas: spaCy, ClearNLP, CoreNLP e MATE. Todas elas têm uma integração simples.

O ClearNLP, atualmente com o nome de NLP4J [35], é uma ferramenta em Java, que fornece funcionalidades NLP, *frameworks* para o desenvolvimento rápido e eficiente de componentes NLP e ainda uma API para a manipulação de estruturas computacionais de NLP, como por exemplo, gráficos de dependências.

MATE é também uma ferramenta em java [36], que fornece suporte para a anotação de discurso e de texto. Inclui funcionalidades que facilitam a exibição e edição dessas anotações, bem como suporta consultas complexas às mesmas. Pode ser usado qualquer esquema de anotação, desde que seja codificado em XML.

O spaCy oferece uma ampla lista de funcionalidades, incluindo *Neural Network models*, *Integrated word vectors*, suporte de várias línguas, *tokenization*, *part-of-speech tagging*, segmentação de frases, *dependency parsing*, reconhecimento de entidades e *entity linking* [37]. É uma biblioteca de código *open source* para processamento avançado de linguagem natural, que permite apresentar respostas a perguntas. tais como, o que é determinado conceito, qual o seu significado num determinado contexto, e que textos são semelhantes entre si.

O spaCy é a única ferramenta implementada em python, o que permite ser usada diretamente pela plataforma, sem a necessidade de recorrer ao uso de APIs. É também a ferramenta que disponibiliza informação mais completa e, segundo artigos revistos em 2015 confirmaram que "*o spaCy oferece o analisador sintático mais rápido do mundo e que sua precisão está dentro de 1% dos melhores disponíveis. Os poucos sistemas mais precisos são 20 vezes mais lentos.*"[37][38]. Esta foi a ferramenta escolhida.

Tabela 3 - Tabela comparativa NLP. Adaptada de: [37]

System	***Year***	***Language***	***Accuracy***	***Speed (wps)***
spaCy ***v2.x***	2017	Python / Cython	**92.6**	*n/a*
spaCy ***v1.x***	2015	Python / Cython	91.8	13,963
ClearNLP	2015	Java	91.7	10,271
CoreNLP	2015	Java	89.6	8,602
MATE	2015	Java	92.5	550

Após alguma análise do que existia, o spaCy foi a ferramenta escolhida, uma vez que oferece as funcionalidades pretendidas com uma baixa curva de aprendizagem e com a vantagem de que permite treinar modelos com dados externos, que vai de encontro com os requisitos definidos.

O primeiro passo foi iniciar um estudo sobre o que já existe sobre o spaCy ser treinado com termos do UMLS. Após alguma análise, foi possível perceber que existem duas formas diferentes adicionar entidades ao spaCy, sendo estas feitas através de uma base de conhecimento (KB – *Knowlegde Base*) ou com o treino de modelos.

Entity Linking é uma das funcionalidades que o spaCy oferece, para executar a ligação de entidades, feito através de uma base de conhecimento (KB – *Knowledge Base*). É recente, foi apresentada em agosto de 2019 na "spaCy IRL *conference*", e ainda está em exploração. Pretende vir resolver o problema de existirem entidades com o mesmo nome. Na conferência, a oradora dá o exemplo de Lord Byron, Lady Byron e Ada Lovelace (as duas filhas dele). Um texto que se refira a qualquer um deles como "Byron", este modelo poderá ser capaz de distinguir a qual das entidades o texto se estará a referir [39].

QuickUML [40] é uma ferramenta para extração de informação de conceitos biomédicos de texto no contexto da medicina. Utiliza *Simstring* para fazer o *match* das *strings*. A ferramenta seleciona o *subset* mais apropriado depois de alguns cálculos. Faz o *tokenizer* e obtém as *tags* de cada *token*. Para isso foi usado o spaCy. Os *tokens* podem ser palavras ou pontuação. Eles permitem a pontuação porque no domínio médico, a pontuação pode fazer parte do nome do medicamento, ou tratamento, ex.: "*Lantus (Insulin Glargine*)". Os autores comparam o QuickUML com o MetaMap e o cTakes, tendo este apresentado melhores resultados.

O ScispaCy [39] é um repositório que contém modelos e *pipes* personalizados e relacionados com o uso do spaCy para documentos científicos. Em particular, existe um *tokenizer* personalizado que adiciona regras baseadas no spaCy, um *analyzer* com dados biomédicos e ainda um modelo de reconhecimento de entidades.

Por último, UMLSEntityLinker é uma *alpha feature* do spaCy que executa a vinculação ao UMLS usada pelo ScispaCy. O *linker* realiza uma pesquisa de sobreposição de *strings* nas entidades, comparando-as com uma base de conhecimento de 2,7 milhões de conceitos usando uma pesquisa de vizinhos mais próximos. No entanto, este componente ainda não é muito robusto, sendo necessário alterar e testar alguns dos parâmetros para se adaptar ao caso de uso.

De modo a obter uma solução completamente adaptada à realidade da plataforma doDOC e com a maior versatilidade possível, será implementado um processo interno que usa o spaCy para extrair a informação com o dicionário do UMLS.

3.3. Repositórios Externos de Informação

Após a escolha do spaCy como ferramenta para a extração de informação, e com o objetivo de construir um modelo de treino, foi necessário efetuar uma pesquisa de possíveis fontes externas, com termos médicos, que possam ser usadas para a construção de um dicionário interno de termos.

UMLS é um projeto mantido pela *U.S. National Library of Medicine* (NLM) que integra e distribui diversas fontes de conhecimento médico (bases de dados). Estas fontes de conhecimento são multiuso, isto é, não são otimizadas para uma aplicação em especifico e podem ser aplicadas em diversos sistemas. Disponibiliza três repositórios de informação: o *Metathesaurus*, o *Semantic Network* e o *Specialist Lexicon and Lexical Tools*.

O *Metathesaurus* é um repositório que contém termos e códigos de vários vocabulários, incluindo:

- O CPT (*Current Procedural Terminology*), mantido para *American Medical Association* (AMA) e que contém uma lista de termos descritivos bem como códigos para a identificação de serviços e procedimentos médicos[41];
- O ICD10CM (*International Classification of Diseases, Tenth Revision, Clinical Modification*), mantido pelo *National Center for Health Statistics* (NCHS) e o *Centers for Medicare and Medicaid Services* (CMS), que disponibiliza uma lista tabular que contém os códigos de doenças, descrições e notações instrucionais associadas[42];
- O LOINC (*Logical Observation Identifiers Names and Codes*), desenvolvido e mantido pelo *Regenstrief Institute, Inc*, que disponibiliza dados que abrangem a terminologia do laboratório, sinais vitais, entrada/saída hemodinâmica,

eletrocardiograma, ultrassom obstétrico, eco cardíaco, urológico, imagiológico, procedimentos gastro endoscópicos, gestão do ventilador pulmonar e instrumentos de pesquisa selecionados[43];

- O MeSH (*Medical Subject Headings*), desenvolvido e suportado pela NLM, que disponibiliza, entre outra informação, dados de químicos suplementares como produtos químicos, protocolos, doenças e organismos[44];
- O RxNorm, uma nomenclatura padrão de medicamentos clínicos para humanos que é produzida para NLM [45];
- O SNOMED CT (*Systematized Nomenclature of Medicine*), cujos direitos de propriedade pertencem ao IHTSDO (*International Health Terminology Standards Development Organisation*), que disponibiliza conceitos de vocabulários em sete áreas diferentes: Topografia, Morfologia, Etiologia, Função, Doença, Procedimento e Ocupação [46].

A lista completa de todos os vocabulários que integram a biblioteca *Metathesaurus* encontra-se disponíveis na documentação disponibilizada pela NLM [47].

O *Semantic Network* disponibiliza a informação de tipos semânticos, ou seja, o seu tipo relativo ao seu significado, e os seus relacionamentos semânticos (relações entre tipos semânticos). O *Specialist Lexicon and Lexical Tools* apresenta um largo léxico sintático de conceitos biomédicos e ferramentas para normalizar sequências de caracteres, gerar variantes lexicais e criar índices[47].

Destes, o *Metathesaurus* é o repositório de informação mais completo relativamente ao tipo de informações que contém para o tipo de dados que se pretende indexar, e por esse mesmo motivo foi o escolhido para ser analisado.

3.4. Full Text Search e Indexação de Dados

O objetivo pretendido nesta secção é encontrar uma ferramenta que permita usar a funcionalidade de *full text search* e indexação de documentos, e, de acordo com o capítulo 2, são tidos alguns requisitos em conta, como por exemplo, ser uma ferramenta integrável com a linguagem Python. A Tabela 4 compara várias ferramentas utilizadas para o objetivo pretendido.

Tabela 4 - Comparação de Ferramentas para *Full Text Search*

Ferramenta	***Descrição***	***Suporta Indexação***	***BD não relacional***	***Linguagem***	***Compatibilidade com Python***	***Funcionalidades (Full Text Search)***
MongoDB	Base de dados não relacional	Sim	Sim	NA	Sim	Indexação granular e dinâmica; Suporta ampla variedade de operadores de consulta; *Analysers*; *Scoring*; *Highlighted snippets*; [48]
Apache Solr	Servidor de pesquisa *open source*, construído sobre o Apache Lucene	Sim	Sim	Java	inclui um *output* específico para Python, mas a resposta em JSON é um pouco mais robusta.[49]	Indexação de *Java Beans*; *Solr Queries*; *Filter Queries*; *Faceted Search*; *Highlighting*; Sugestões de Pesquisa; *Spell Checking*;
Whoosh	Biblioteca para sistemas de pesquisa	Sim	Sim	Python	Sim	Indexação de Documentos; *Highlighting*; *Parsing* de *user queries*; *Analysers*; *Stemming;* *Sorting* e *faceting*; *Concurrency* e *locking*;
Scout	Servidor restful baseado no SQLite	Sim	Não	Python	Sim	índices de pesquisa presentes numa base de dados; Autenticação baseada em chave simples (opcional). Armazena conteúdo de pesquisa e os metadados Vários algoritmos de classificação de resultados Testes unitários abrangentes. [50]
Elastic Search	Sistema para pesquisa e análise, baseado no Apache Lucene	As indexações são mais lentas	Sim	NA	Sim	Pesquisa *Fuzzy*; permite um controlo da ordem e proximidade dos termos; Suporta a sintaxe do Lucene;
Sphinx	Sistema de pesquisa *open source* para pesquisar em dados de diferentes fontes	Sim	Sim	C++	Dispõe de uma API compatível	alto desempenho de indexação e pesquisa; *tokenizer* de texto flexível, vários modos de classificação; pós-processamento avançado de conjunto de resultados; [47]
Python Stdnet	Mapeador de objetos de dados e gestor de consultas avançadas para base de dados	Sim	Sim	Python	Sim	API de consulta avançada; Conexão de bases de dados síncrona e assíncrona;
Lucene	Biblioteca que facilita a adição de funcionalidades de pesquisa a uma aplicação	Sim	Sim	Java	Biblioteca PyLucene permite aceder aos mesmos recursos	*Input Stream*; *Tokenizer*; Filtro *Lower Case*; Filtro *StopWord*; *Similarity* e *Scoring*;

De entre as várias ferramentas potenciais desenvolvidas Python, destaca-se o PyLucene pelas funcionalidades que oferece e também pelo número de sistemas aplicacionais que a usam como ferramenta base.

Em termos de integração com a plataforma doDOC, o ElasticSearch seria a menos fácil de integrar, pois dado que funciona como um motor de base de dados, obrigaria a que as estruturas de dados a serem indexadas/pesquisas fossem exportadas para esse motor.

Apesar da análise destas soluções, foi decidido a utilização do MongoDB, uma vez que este não requer nenhuma integração adicional, pois é uma das ferramentas usadas atualmente na plataforma, e que oferece as funcionalidades essenciais para *Full Text Search*.

O MongoDB é uma base de dados *open source*, escalável e orientado a documentos. As bases de dados orientadas a documentos substituem o conceito de "*row*" por um modelo mais flexível: "*document*". Este modelo permite a utilização de documentos embutidos e *arrays*, possibilitando o uso de estruturas hierárquicas mais complexas. O facto de ser escrito na linguagem C++, faz com que seja portável para diferentes sistemas operativos [51].

De entre as funcionalidades do MongoDB, é possível destacar a indexação, e o *Full Text Search* que permite pesquisar texto de uma base de dados. Este mecanismo de pesquisa examina todas as palavras de modo a corresponder ao texto que o utilizador especificou. O MongoDB suporta operações de consulta que executam uma pesquisa do conteúdo de uma *string*, através do uso de indexação de texto, que podem incluir qualquer valor que seja uma *string* ou um *array* de *string*s.

Após a criação de um *index* de texto a uma coleção, o MongoDB permite a pesquisa de texto completa (*full text search*) através do operador "$*text*". Este operador, faz um *tokenize* da *string* pesquisada, usando como delimitadores os espaços em branco e a pontuação existente, seguido da aplicação do operador lógico OR a todos os *tokens* da *string*.

No contexto do protótipo, o MongoDB introduz uma limitação ao seu desenvolvimento: a impossibilidade de fazer pesquisas com palavras parciais, isto é, para usufruir das vantagens do *Full Text Search*, as palavras que o utilizador inserir, terão de estar completas [52].

O MongoDB oferece outras funcionalidades, tais como;

- *Stop words*, ou seja, ignora *stop words* especificas de cada língua, como por exemplo "*the*" e "and" em inglês;
- *Stemmed words*, por exemplo, se um documento contiver a palavra *"blueberry"* e o termo de pesquisa for *"blue"* não irá fazer *match*, no entanto, *"blueberries"* irá;
- *Case sensitive* e *case insensitive*;
- *Text score*, em que é atribuída uma pontuação a cada documento que contem o termo de pesquisa nos campos indexados. Esta pontuação representa a relevância de um documento para uma determinada pesquisa. Os resultados podem ser ordenados de acordo com este score;

- Pesquisas por frase exata, bastando para isso que a pesquisa seja colocada entre aspas, e os resultados serão encontrados com match da frase na integra;
- É possível fazer uma pesquisa que exclua termos;
- *Case and Diacritic Insensitive Search*, ou seja, o índice não faz distinção entre caracteres que contêm pontuação diacrítica;
- Limitar a pesquisa com um número definido.

O MongoDB suporta várias línguas, no entanto, no contexto do protótipo será apenas utilizada a língua inglesa.

De acordo com a análise efetuada das soluções/ferramentas existentes, além de complexas, não satisfazem os requisitos definidos (capítulo 2).

Deste modo, e tomando em consideração ser possível a integração de alguns repositórios relevantes, como o UMLS, e para que possam ser integradas outras fontes de dados, optou-se por definir uma proposta de extração de informação.

4 Proposta de Extração e Integração de Informação

O presente capítulo apresenta uma proposta do sistema de extração, processamento e integração de informação de repositórios externos e descreve alguns aspetos relacionados com a implementação do protótipo desenvolvido. A proposta, ilustrada na Figura 9, contempla cinco fases:

- Recolha/integração de repositórios de informação externos;
- Integração no MongoDB nas estruturas específicas;
- Treino do modelo para aprender a identificar esses termos;
- Criação da estrutura para indexação;
- Desenvolvimento de um *endpoint* para retorno dos resultados mediante uma *query* do utilizador.

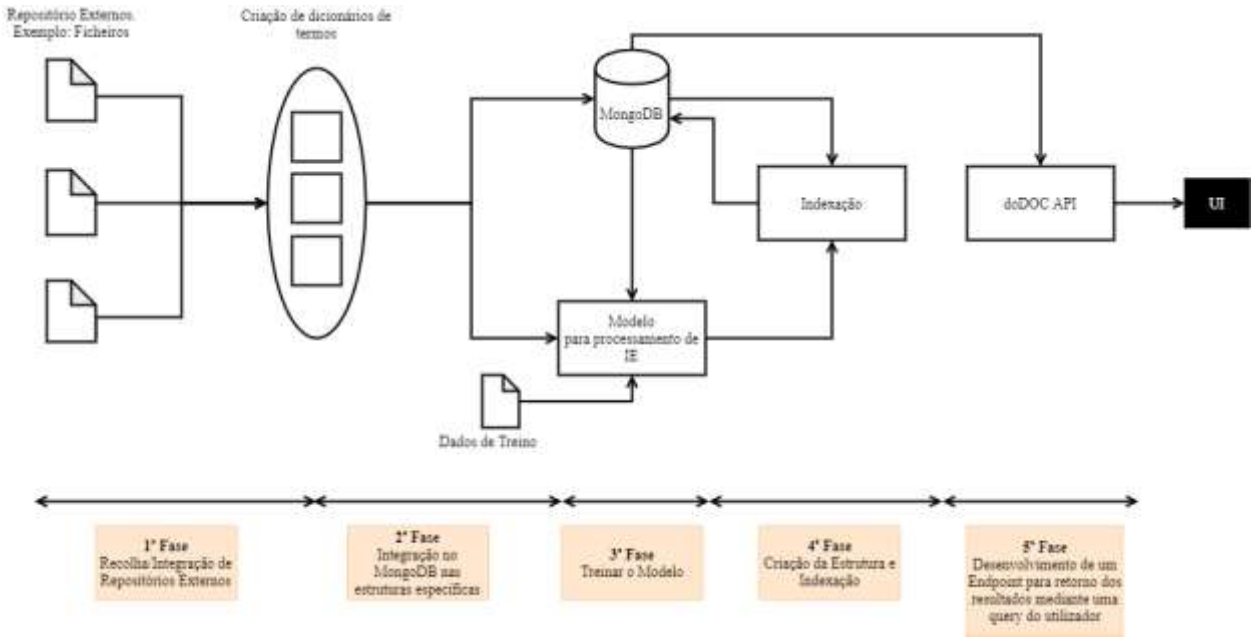

Figura 9 - Arquitetura da Solução Proposta

As secções seguintes descrevem cada uma destas fases.

4.1. Recolha/Integração de Repositórios Externos (1ª Fase)

A recolha e integração de repositórios externos depende do âmbito ou contexto para o qual se pretende construir um novo dicionário para a plataforma doDOC. Após a definição do âmbito pretendido, procede-se ao estudo e avaliação dos potenciais repositórios de dados, provenientes de fontes fidedignas e que forneçam informações para o contexto, como a definição dos termos. Na seleção do(s) repositório(s) é relevante analisar e compreender as estruturas de dados que guardam os termos e as suas ligações/relações, por forma a ser possível integrar numa estrutura *standard*.

Para o protótipo, que se insere no contexto dos *clinicals trials*, foram identificados alguns repositórios de dados (secção 3.3. Repositórios Externos de Informação) que contêm informação relevante e que podem ser extraídas. Esta secção descreve o processo para extrair a informação do repositório de informação externa.

O Repositório Metathesaurus

Para a identificação da fonte a usar no protótipo foi feita uma pesquisa que resultou na escolha do UMLS, especificamente no repositório *Metathesaurus*, uma vez que, é o repositório mais completo relativamente ao tipo de informações que contem para o tipo de dados que se pretende anexar.

O *Metathesaurus* é o maior componente do UMLS e contém conceitos, nomes de conceitos e outros atributos de mais de 100 terminologias, classificações e léxico. Um dos principais objetivos é conectar diferentes nomes para o mesmo conceito [53].

Numa primeira abordagem, foi necessário perceber o conteúdo dos ficheiros do *Metathesaurus*.

Estes ficheiros distinguem-se em várias categorias. A primeira, lista os conceitos, nomes de conceitos e as suas fontes. A segunda, descreve os atributos, incluindo todas as informações discretas sobre os conceitos, átomos ou relações. Átomos são os nomes dos conceitos ou as *strings* de cada um dos vocabulários de origem. Isto é, para o mesmo conceito podem existir vários átomos, mediante o seu vocabulário de origem. A Tabela 5 mostra um exemplo de um conceito representado de várias formas (vários átomos) em vocabulários diferentes. Atributos de conceitos são adicionados durante a construção do *Metathesaurus* e aplicam-se a todos os nomes do conceito. Por exemplo, os tipos semânticos "*Pathologic Function*" e "*Finding*" são atributos do conceito "*Atrial Fibrillation*" e são aplicáveis a qualquer átomo conectado a esse conceito. Atributos de átomos pertencem a um vocabulário de uma fonte de origem específica. Alguns são de interesse geral, outros são relevantes apenas para um vocabulário. Por exemplo, a definição "*Disorder of cardiac rythm characterized by rapid, irregular atrial impulses and ineffective atrial contractions*" é um atributo do átomo "*Atrial Fibrillation*" proveniente da fonte *Medical Subject Headings* (MeSH).

Tabela 5 - Exemplos de átomos no *Metathesaurus*. Adaptado de: [54]

Term	***Source Vocabulary***
Atrial fibrillation	ICD-9-CM
AF	NCI Thesaurus
AFib	MedDRA
Atrial fibrillation (disorder)	SNOMED Clinical Terms
atrium; fibrillation	ICPC2-ICD10 Thesaurus

De acordo com as categorias do UMLS, existem ficheiros no *Metathesausus* que representam tabelas de uma base de dados relacional. A Tabela 6 mostra os ficheiros existentes para cada categoria.

Tabela 6 - Ficheiros *Metathesaurus*. Adaptado de: [54]

Metadata File Name	***Contents***
MRCONSO.RRF	*Names, Synonyms, Terms, Term Types, Codes*
MRREL.RRF	*Relationships*
MRHIER.RRF	*Hierarchies*
MRSAT.RRF	*Attributes*
MRDEF.RRF	*Definitions*
MRMAP.RRF	*Mappings*
MRSMAP.RRF	*Simplified Mappings*
MRSTY.RRF	*Semantic Types*

Como estes ficheiros contêm muitos dados para analisar, a estratégia usada foi analisar todos os atributos dos mesmos (que correspondem aos *headers* das colunas da base de dados), selecionar os que têm maior potencial para o desenvolvimento do sistema proposto e selecionar os ficheiros a que os mesmos pertencem. O Apêndice A contém uma tabela que lista esses atributos, bem como a sua definição e o ficheiro onde se encontram.

Após ter um levantamento dos atributos mais relevantes, foram analisados os ficheiros em que estes se encontravam, de forma a perceber o seu propósito. A Tabela 7 apresenta uma descrição desses ficheiros.

Tabela 7 - Listagem de ficheiros *Metathesaurus* [53]

Ficheiro	*Descrição*
MRAUI.RRF	Regista o movimento de um AUI (*Atom Unique Identifier*) de um conceito para um CUI (*Concept Unique Identifier*) na versão seguinte do *Metathesaurus*
MRCUI.RRF	Neste ficheiro há uma ou mais linhas para cada CUI, que existiam em qualquer *release* anterior, mas que já não estão presentes na *release* atual
MRMAP.RRF	Representa conjuntos de mapeamentos entre vocabulários. Esse mapeamento ocorre entre identificadores de dois vocabulários diferentes
MRHIER.RRF	Este ficheiro contém uma linha para cada hierarquia ou contexto em que cada átomo aparece
MRREL.RRF	Existe uma linha nesta tabela para cada relacionamento entre átomos ou conceitos conhecidos no *Metathesaurus*
MRSAT.RRF	Há exatamente uma linha nesta tabela para cada conceito, átomo ou atributo de relacionamento que não possui uma estrutura de subelementos
MRDEF.RRF	Descreve as definições no *Metathesaurus*. Uma definição é um atributo de um átomo
MRDOC.RRF	Este ficheiro contém uma linha para cada valor permitido dos elementos ou atributos de dados selecionados que possuem um número finito de abreviações como valores permitidos
MRSTY.RRF	Neste ficheiro existe uma linha para cada tipo semântico atribuído a cada conceito.
MRCONSO.RRF	Este ficheiro contém uma linha para cada átomo, contendo cada nome, *string* ou conceito no *Metathesaurus*

Após a análise dos ficheiros, e dos seus respetivos dados, foi necessário identificar quais os que continham informações relevantes, e que deveriam ser utilizados para a criação do dicionário interno. Por exemplo, apesar do ficheiro MRMAP.RRF conter atributos que são considerados relevantes, como o atributo RELA que representa a relação entre dois conceitos, o propósito deste ficheiro é identificar os diferentes vocabulários para o mesmo conceito, o que, para o objetivo deste protótipo, não era relevante.

Posto isto, dos ficheiros selecionados, foram armazenados os seus identificadores, do conceito ou do átomo, para que informação de diferentes ficheiros pudesse ser cruzada. Foram também armazenados, do ficheiro MRCONSO.RRF os campos foram: LAT - para ter a informação sobre a língua do termo, TTY, para saber qual o tipo do termo e STR, o nome do termo. O ficheiro MRDEF.RRF foi usado para obter o campo DEF, que contém a definição do conceito. Do

ficheiro MRSTY.RRF foi guardado o campo STY – que define o tipo de semântica do conceito. Do ficheiro MRREL.RRF, foi guardado o campo RELA. Por último, do ficheiro MRHIER.RRF, foram guardadas as informações dos campos RELA, do AUI e do PAUI (AUI do conceito pai em relação ao outro).

Para o efeito, os dados foram carregados dos ficheiros para uma base de dados em MySQL. Devido à dificuldade de carregar ficheiros tão pesados, sendo que o maior tinha aproximadamente 7 gigabytes, foram aplicadas primeiramente algumas operações na linha de comandos. Do ficheiro MRCONSO.RRF foram filtradas da tabela apenas as colunas necessárias e os termos que se encontravam em inglês. Do ficheiro MRREL.RRF foram filtradas também as colunas necessárias e os termos cujo AUIs estavam presentes no ficheiro MRCONSO já filtrado. O mesmo processo foi usado para filtrar os dados do ficheiro MRHIER.RRF.

Após este passo, foi possível construir algumas *queries* de forma a agrupar a informação desejada. Abaixo, encontra-se um exemplo de uma das *queries* utilizadas, neste caso, usada para guardar as hierarquias dos termos.

```
SELECT
     conso.str AS 'name',
     conso.tty AS short_type,
     terms.description AS 'type',
     hier.rela AS relation,
     conso2.str AS name2,
     conso.lat AS 'language'
FROM
     metathesaurus.mrhier AS hier,
     metathesaurus.mrconso AS conso,
     metathesaurus.mrconso AS conso2,
     metathesaurus.term_types AS terms
WHERE
     rela is not NULL
     and hier.aui = conso.AUI
     and hier.PAUI = conso2.AUI
     and conso.tty = terms.short_term;
```

Da *query* apresentada, podemos retirar como exemplo a seguinte relação:

"Persistent depressive disorder (dysthymia) **is a** Depression."

Neste caso, existem dois conceitos identificados: "Persistent depressive disorder (dysthymia)" e "Depression" que se encontram na tabela HIER, ligados pela relação "is a".

Inicialmente os dados foram carregados e trabalhados em MySQL, no entanto foram encontrados alguns problemas de performance ao cruzar todos os dados, mesmo após a indexação dos campos usados. Sendo que o resultado seria usado na base de dados não relacional, usada na plataforma doDOC (MongoDB), decidiu-se fazer a migração destes dados para a mesma.

4.2. Integração no MongoDB (2ªfase)

A integração no MongoDB passa pela criação de uma estrutura *standard* que permita armazenar termos e informações relevantes sobre estes, provenientes do repositório de informação escolhido, de forma a criar um dicionário interno com termos de um determinado âmbito. Alguns dos dados que cada entrada na estrutura guarda são: um identificador relacionado com a sua fonte, a língua em que se encontra, o tipo semântico do termo, a sua definição e as suas relações. Dentro das relações, a informação que pode ser guardada é o tipo da relação, o termo relacionado e o identificador. De todos estes campos, os únicos obrigatórios são o termo e o seu tipo semântico, de forma a existir flexibilidade entre fontes diferentes, podendo variar, de acordo com a área onde se focam.

Para o protótipo, foi criado um dicionário interno relacionado com termos médicos e foram usados alguns exemplos de *clinical trials* relacionados com a doença "*depression*". Esta escolha foi feita para limitar a recolha de termos, uma vez que a fonte tem imensos dados, que dificultariam os testes feitos ao protótipo.

O processo para a construção está ilustrado no fluxograma da Figura 10. O processo começa com a leitura de um ficheiro que contem informação de alguns *clinical trials* em formato de texto. Nesse ficheiro, é aplicado um processo de *tokenization* para que as palavras possam ser analisadas individualmente. Neste ponto, é analisado se, cada uma dessas palavras se encontra na coleção criada na secção anterior: `MRCONSO`. `MRCONSO` é a coleção que corresponde ao ficheiro do *Metathesaurus* que contem cada nome, *string* ou conceito e a língua no *Metathesaurus*. É a partir deste ficheiro que se irão procurar o resto das informações, como a definição do termo e as suas relações. Para cada termo encontrado no ficheiro que coincide com o termo da coleção `MRCONSO`, é criada uma entrada no dicionário interno, sendo guardado o nome do termo, a sua língua e o identificador que o *Metathesaurus* usa para identificá-lo.

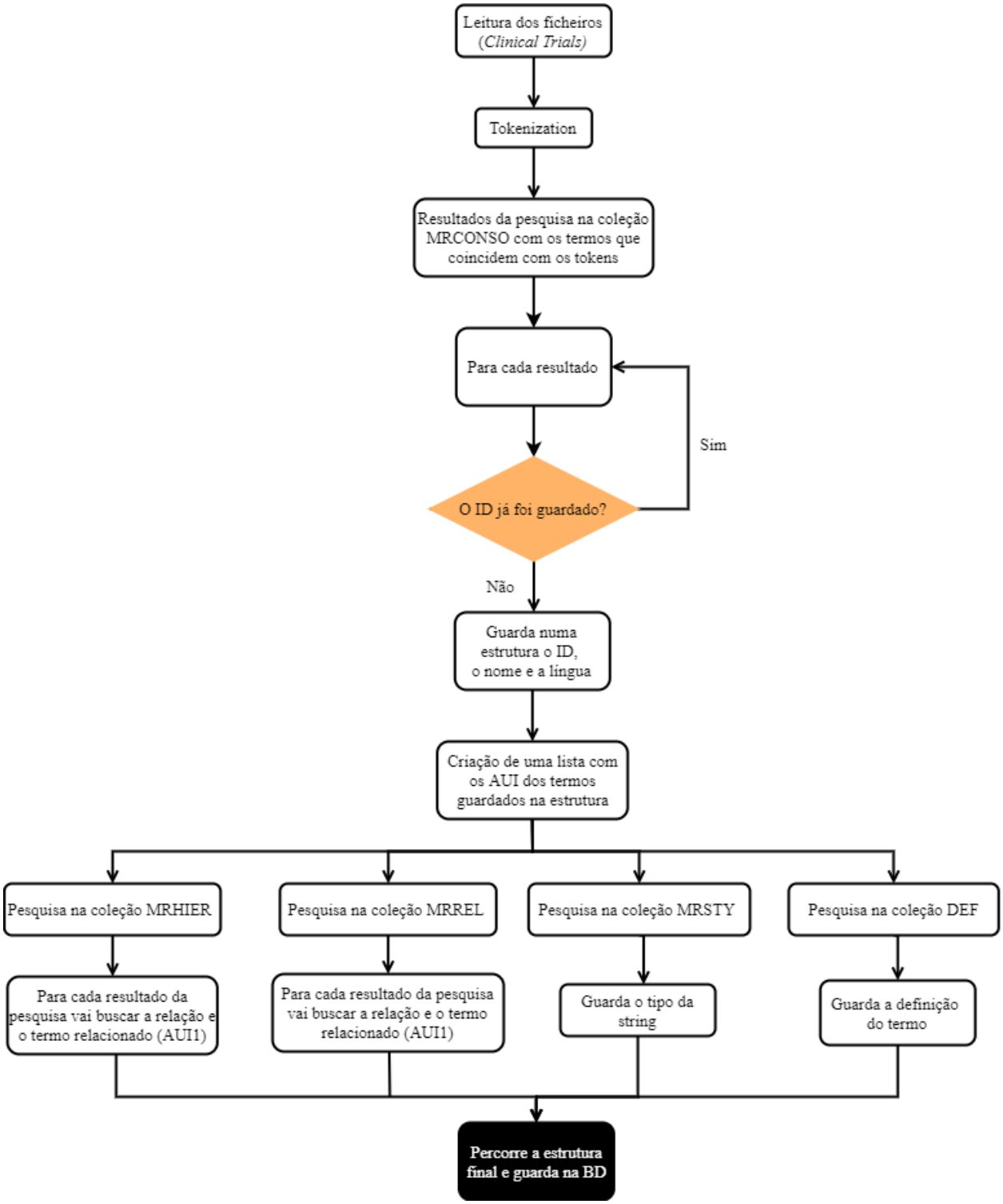

Figura 10 - Fluxograma para a construção do conjunto de dados

Depois de os termos terem sido todos percorridos e guardados, a partir do seu identificador, são mapeados os dados com as outras coleções. Isto é, na coleção `MRHIER` é possível encontrar as relações de hierarquia relacionadas com o termo. Todas elas são guardadas numa lista associada ao termo. Na coleção `MRREL`, encontram-se outros tipos de relações, como por exemplo, "*is_associated*" ou "*has_component*". Nestes dois casos em que são encontradas

relações (provenientes do `MRHIER` ou do `MRREL`), estas são guardados, juntamente com um segundo termo, que é associado através da relação ao termo anteriormente encontrado. Ou seja, se existe uma relação, existem dois termos associados à mesma. Nestas relações, é ainda guardado o identificador do termo relacionado, e a fonte desta relação (a coleção onde foi encontrada). Neste caso, as opções possíveis seriam "*rela*" (proveniente da coleção `MRREL`) ou "*hier*" (proveniente da coleção `MRHIER`).

Por fim, na coleção `MRSTY` e na coleção `MRDEF` são encontrados para o termo, o tipo semântico e a definição do mesmo, respetivamente. A Figura 11 ilustra este mapeamento, isto é, ilustra as coleções de onde são retiradas as informações (lado esquerdo da figura), com o objetivo de serem agregadas na estrutura final, também esta ilustrada na figura.

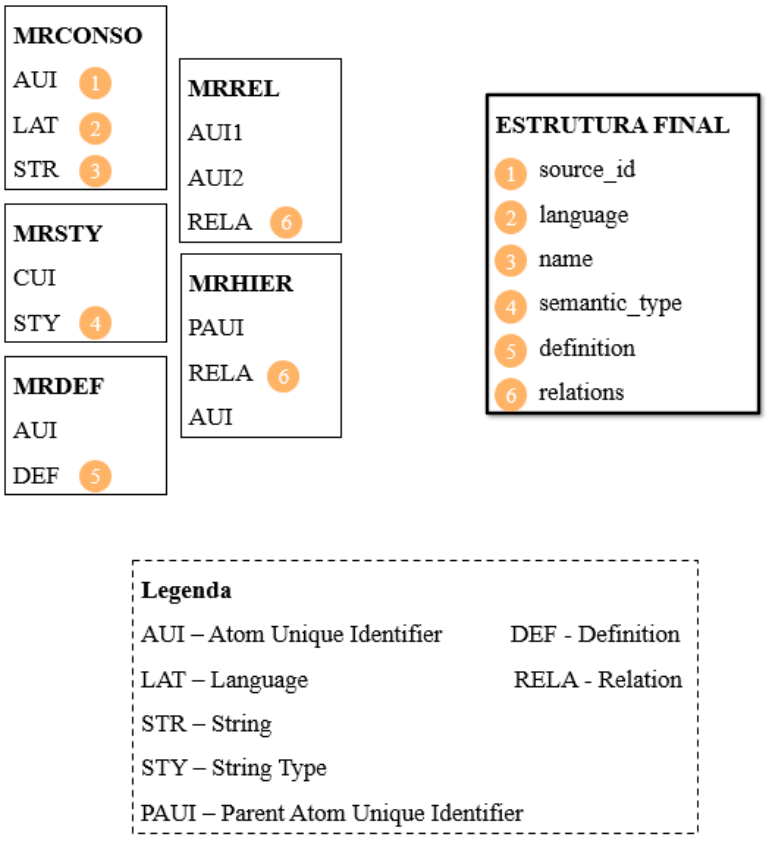

Figura 11 - Mapeamento dos atributos para o dicionário interno

O JSON abaixo representa um exemplo de uma entrada do dicionário criado. O JSON *schema* relativo a este exemplo, que permite uma melhor perceção dos dados e do seu tipo, encontra-se no Apêndice B.

```
{
    "_id"           : ObjectId("5e68226a80357305b24ea6a4"),
    "source_id"     : "A0477546",
    "language"      : "ENG",
    "name"          : "depression",
    "semantic_type" : "Mental or Behavioral Dysfunction",
    "definition"    : "Depressive states usually of moderate
                       intensity in contrast with major depression
                       present in neurotic and psychotic disorders.",
    "relations"     : [
        {
            "relation"   : "associated_with",
            "child_id"   : "A0488119",
            "source"     : "hier",
            "child_name" : "reactive depression"
        }
    ]
}
```

No exemplo acima, para o termo "*depression*", foi guardado o seu tipo semântico: "*Mental or Behavioral Dysfunction*", a língua, a definição e ainda a relação, que neste caso transmite a informação de que o termo "*reactive depression*" esta hierarquicamente ligado ao termo "*depression*". Como esta relação foi encontrada na coleção `MRHIER`, o *source* da mesma é: "*hier*".

4.3. Treino do Modelo (3ªfase)

Após a construção do dicionário com as definições e correlações dos termos clínicos extraídos nas fases anteriores, procede-se ao treino do modelo de processamento para extração de informação dos documentos. Das soluções analisadas (secção 3.2) optou-se pela utilização do spaCy, uma biblioteca *open source* para processamento avançado de linguagem natural, implementado em Python, que apresenta uma baixa curva de aprendizagem e de fácil integração com a arquitetura usada na plataforma doDOC.

O spaCy disponibiliza um conjunto de modelos para diferentes línguas, cujo vocabulário é baseado em texto escrito na web, incluindo blogs, *news* e *comments*[55]. Utilizou-se o modelo "en_core_web_sm", um modelo *multi-task* CNN treinado em inglês no OntoNotes, atribuindo vetores de *tokens* específicos no contexto, *tags*, análises de dependências e entidades. O spaCy possibilita o treino dum modelo a partir de um modelo vazio, ou a partir dum já existente em que são adicionadas novas entidades. Esta segunda abordagem é benéfica pois inclui outras entidades para além das que serão acrescentadas no treino do modelo. Estes modelos linguísticos permitem interpretar palavras da língua, determinar entidades, morfologia (nomes, verbos, conjunções, etc.) e relacionar sintaticamente as palavras encontradas, como por

exemplo sujeitos e objetos. No entanto, é essencial que estas sejam adequadas ao contexto em que irá ser utilizado.

O processo utilizado é igual para todos os tipos de documentos, isto é, qualquer documento da plataforma pode ser usado como *dataset* do modelo para contribuir para o seu refinamento. Neste caso, no contexto dos *clinical trials*, é essencial preparar e treinar o modelo com termos e/ou terminologias adequadas ao mesmo. Os *clinical trials* têm uma vasta gama de termos, condições e doenças. Para o efeito, o modelo criado para o protótipo foi treinado para o contexto "*depression*".

Posto isto, foram selecionados 25 *clinical trials* relacionados com a doença "*depression*", os quais foram divididos aleatoriamente em partes desiguais para o treino do modelo e para avaliação do mesmo. Para facilitar o processo, estes documentos foram convertidos em formato de texto.

Tendo como base os ficheiros de texto, o processo começa uma anotação automática dos termos, identificando se cada um dos termos existentes, em cada um dos ficheiros, se encontra no dicionário criado (secção 4.2). Em caso afirmativo, é feito um mapeamento desse termo com o tipo semântico definido no dicionário, juntamente com a frase onde o termo se encontra, utilizada para dar contexto à entidade.

Tomando como exemplo a seguinte frase de um dos documentos:

"Lifestyle Medicine for Depression"

Para cada um dos elementos sublinhados existe no dicionário uma correspondência, pelo que é criada uma entidade que classifique o elemento de acordo com a classificação semântica do dicionário, de modo a que possa ser interpretada pelo spaCy.

Assim, para a frase foram criadas 3 entidades que foram expressas de acordo com o formato usado pelo spaCy, conforme ilustrado na Figura 12.

```
[
   ('Lifestyle Medicine for Depression', [(0, 9, 'Social Behavior')]),
   ('Lifestyle Medicine for Depression', [(10, 18, 'Biomedical Occupation or Discipline')]),
   ('Lifestyle Medicine for Depression', [(23, 33, 'Mental or Behavioral Dysfunction')])
]
```

Figura 12 – Exemplo de Dados de Treino - spaCy

Para recriar esta estrutura, é identificado o termo (através do seu índice na frase), o seu tipo semântico de acordo com o dicionário criado na secção anterior, e a frase onde este se encontra. Por exemplo, é possível observar que o termo "*Lifestyle*", que se encontra entre os índices 0 e 9 da frase, corresponde à entidade "*Social Behavior*" (tipo semântico). Outro exemplo, o termo *"depression"*, que se encontra entre os índices 23 e 33 da frase, é classificado semanticamente como "*Mental or Behavioral Dysfunction*", e que, se traduz na terceira entrada do conjunto de entidades da Figura 12.

Com o conjunto de dados de treino criado, foi possível efetuar o treino do modelo, de acordo com a documentação disponibilizada pelo spaCy. Utilizou-se o *Minibatch Gradient Descent*, a variante mais recomendada do algoritmo *Gradient Descent* para a maioria das aplicações. Este divide o conjunto de treino em pequenos *batches* que são usados para calcular o erro do modelo e atualizar os seus coeficientes, fazendo com que o modelo faça previsões sobre os dados de treino. As etapas são começam com números menores e vão aumentando gradualmente.

Para cada iteração é calculado o valor do erro do reconhecimento das entidades encontradas. Esse valor é usado para atualizar o modelo, de forma a reduzir esse erro. Para a atualização do modelo, é usado ainda, o parâmetro *dropout*, que representa a probabilidade de uma amostra ser descartada, sendo que, quando mais alto é o seu valor, mais difícil é, para o modelo, memorizar os dados.

Ainda para o treino do modelo, foram usadas duas variáveis. O número de *epochs* e o número de *iterations*. Uma *epoch* representa o processamento da totalidade do conjunto de dados de treino. Isto é, numa *epoch* todos os dados são percorridos. O número de *iterations* corresponde às partições feitas numa *epoch*, sendo o *batch_size* é ajustado em conformidade. Por exemplo, se um conjunto dados for composto por 2000 exemplos divididos em lotes de 50 *(batch_size=50)*, serão necessárias 40 *iterations* para percorrer uma *epoch* [56].

A dificuldade encontrada neste ponto, foi a necessidade de existir contexto para os dados de treino. Isto é, para um treino eficaz do spaCy, é preciso usar exemplos/frases reais. Caso contrário, a qualidade da precisão do modelo é baixa, pois as entidades treinadas não são do mesmo contexto das entidades testadas.

Posto isto, para a avaliação do modelo, foi usado um módulo do spaCy chamado de *Scorer*. Este módulo permite calcular avaliações sobre determinados dados, retornando, para um conjunto de exemplos, os três índices de avaliação utilizados no NER, ou seja, o *precision rate* (P), o *recall rate* (R) e o *F-score* (F), cujo cálculo está ilustrado nas fórmulas seguintes (adaptadas de: [57]). O *F-score* é calculado através da média harmónica do *precison rate* e do *recall rate*, sendo considerado uma avaliação abrangente dos resultados de teste. Quanto maior o *F-score*, melhores são os resultados experimentais [57].

$$P = \frac{the\ number\ of\ entities\ identified\ correctly}{the\ number\ of\ entities\ identified}$$

$$R = \frac{the\ number\ of\ entities\ identified\ correctly}{the\ number\ of\ entities\ presented\ in\ the\ test\ set}$$

$$F_score = \frac{P \times R \times 2}{P + R}$$

Os resultados apresentados na Tabela 8 foram obtidos iterativamente, isto é, após a avaliação de calculada pelo *Scorer* a um modelo treinado, foram treinados outros modelos com parâmetros ajustados mediante os resultados que foram obtidos na avaliação feita pelo *Scorer*. Para chegar a estes resultados foi necessário seguir diferentes abordagens e corrigir alguns erros/obstáculos encontrados, que estavam a propiciar valores baixos na avaliação dos modelos.

O primeiro obstáculo encontrado foi na construção dos dados de treino e de testes (de acordo com o formato utilizado pelo *spaCy*). Conforme ilustrado na Figura 12 para uma frase que contém 3 entidades diferentes, foram criados respetivamente 3 exemplos. Posteriormente, percebeu-se que essas 3 entidades deveriam estar agregadas no mesmo exemplo, ao invés de produzir 3 distintos. A Figura 13 ilustra o mesmo exemplo da Figura 12, mas com as entidades agregadas.

```
[
    ('Lifestyle Medicine for Depression',
            [
                    (0, 9, 'Social Behavior'),
                    (10, 18, 'Biomedical Occupation or Discipline'),
                    (23, 33, 'Mental or Behavioral Dysfunction')
            ]
    ),
]
```

Figura 13 - Exemplo de Dados de Treino (melhorado)

A representação inicial induziu a uma perceção irreal do número total de exemplos que estavam a ser considerados para treino, pois o número de frases a ser consideradas nas interações era variável e muito superior ao número real de frases.

Outro obstáculo encontrado derivou da identificação de algumas preposições que se encontram no dicionário interno, proveniente do modelo base utilizado ("en_core_web_sm"). Por exemplo, a preposição "*in*" está no dicionário interno, e, por isso, é identificada como sendo uma entidade do tipo "*Geographic Area*". Este fator também influência no número de exemplos e a relevância para o contexto do protótipo. Por esse mesmo motivo, estas preposições não foram contempladas no dicionário.

Para a avaliação dos modelos, foram também analisadas duas abordagens diferentes para o valor *dropout* (probabilidade de cada frase/exemplo ser descartado). A primeira abordagem passou por usar um valor iterativo para o *dropout*. Para isso foi usada a função *decaying* que o *spaCy* disponibiliza, e que utiliza nos exemplos. Esta função produz uma série infinita de valores em decomposição linear. Os valores usados foram 0.6 e 0.2, o que significa que, no treino, o *dropout* começava com um valor de 0.6, que decrescia no máximo até 0.2. O valor do *decaying factor* variou de 0.001 e 0.0001, que determinouo intervalo em que o *dropout* desce. A outra abordagem usada foi a utilização do valor estático para o *dropout*: 0.35.

Tabela 8 - Avaliação dos modelos de treino – spaCy

Modelo	***Dados de Treino***			***Avaliação do Modelo***				
	Epoch	*Iterations*	*Dropout*	*Tempo de Execução (horas)*	*F-score*	*Precision*	*Recall*	*Nº de Entidades*
M1	15	100	Decay: 0.001	15.36	81.92	85.71	94.56	34/35
M2	5	100	Decay: 0.001	3.07	91.23	89.81	92.68	33/35
M3	5	100	Decay: 0.0001	3.64	90.49	89.98	90.99	27/35
M4	5	100	0.35	2.61	92.16	89.56	94.93	34/35
M5	3	200	0.35	3.08	91.04	88.77	93.43	33/35
M6	3	100	0.35	1.99	91.71	90.05	93.43	32/35
M7	**5**	**50**	**0.35**	**1.81**	**92.90**	**91.30**	**94.56**	**34/35**
M8	5	10	0.35	0.45	83.04	86.41	79.92	24/34

O objetivo do treino dos modelos M1 e M2 foi comparar o impacto da variação do número de *Epochs*, com os mesmos valores para as *iterations* e para o *dropout.* Com 5 *epochs* os resultados obtidos foram melhores. Por esse motivo, e também porque o tempo de execução é muito elevado, não foram feitos mais treinos com esse valor de *epochs.*

Depois, para 5 *epochs* e mantendo o número de *iterations* em 100, foi comparada as duas abordagens diferente para o *dropout.* Primeiramente com um valor iterativo (*Decaying)* e depois com o valor estático, 0.35 (valor recomendado na documentação do *spaCy).* Os resultados obtidos estão representados nos modelos M3 e M4 e apresentam um valor de *f-score* de 90.49% e 92.16%, respetivamente. O que indicia que o valor estático se adequa melhor para os dados de treino avaliados, além de apresentarem um menor tempo de execução.

No modelo M6, mantiveram-se as 100 *iterations* e o valor de 0.35 no *dropout* e diminuiu-se o número de *epochs* para 3. O que, não resultou numa melhoria no valor do *f-score*. Testou-se também com 3 *epochs*, mas desta vez com 200 *iterations* (modelo M5) e o resultado também não melhorou.

Nos modelos M7 e M8, mantiveram-se os parâmetros que, até esta altura representaram melhores resultados, isto é, 5 *epochs* e 0.35 no valor do *dropout* e diminui-se o número de *iterations*, e consequente aumento do tamanho do *batch_size*, uma vez que, para efeitos de teste, o volume de dados não é muito extenso, e pretendia-se encontrar um compromisso entre a precisão e o tempo de execução.

Como ilustrado no gráfico da Figura 14, o modelo M7 foi o modelo que conseguiu atingir um valor melhor de F-score, sendo este de 92.9%. Para obter este resultado, foram usadas 5 *epochs* e 50 iterações.

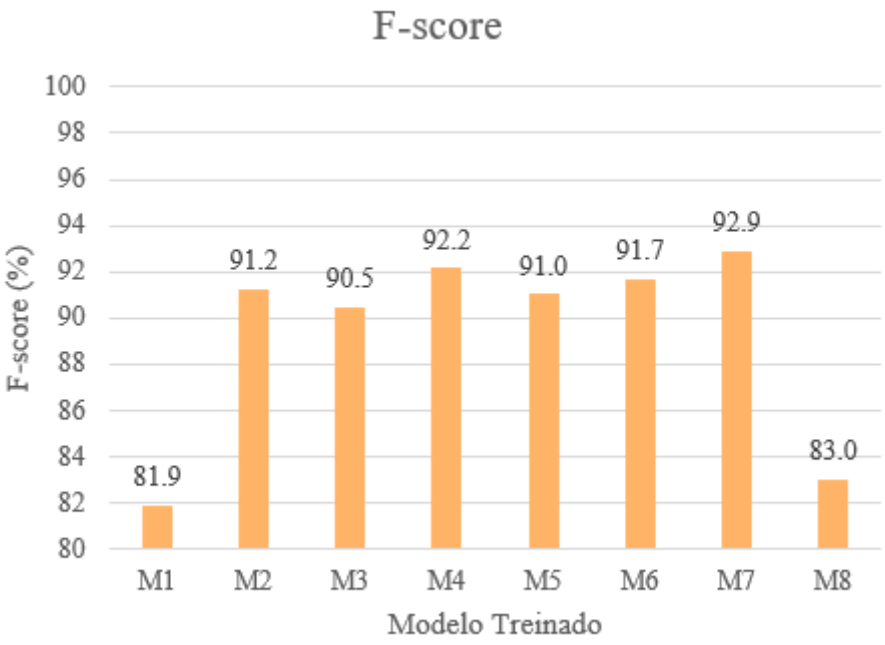

Figura 14 - Comparação de F-score dos Modelos

O tempo de execução para o modelo M7 (ilustrado no gráfico da Figura 15), foi de 1.81 horas, isto é, 1 hora e 49 minutos, o que, comparativamente com os tempos dos outros modelos, é um tempo de execução baixo. De todos os modelos treinados, o M8 é o único que demorou menos tempo a ser treinado, no entanto os resultados que apresenta são também eles mais baixos.

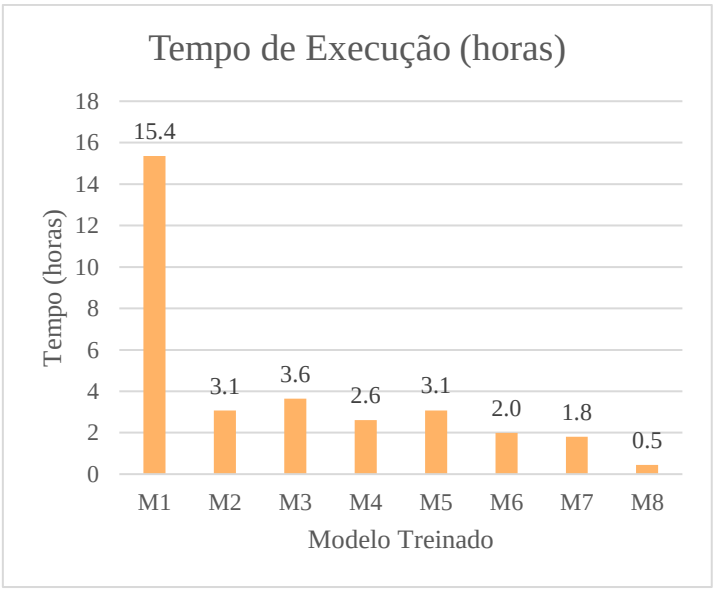

Figura 15 - Comparação do Tempo de Execução dos Modelos

Quanto ao número de entidades encontradas, o número total de identidades presentes nos dados de teste é 35. A Figura 16 indica o número de entidades encontradas por cada um dos modelos. O melhor resultado obtido, foi a identificação de 34 das 35 entidades, sendo que, também neste valor, o modelo M7 conseguiu obter bons resultados.

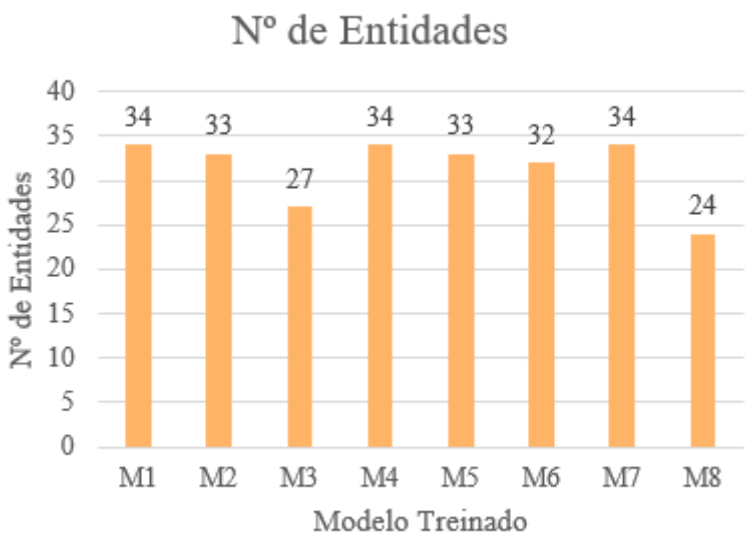

Figura 16 - Nº de Entidades Encontradas pelos Modelos

Foi também efetuada uma avaliação dos valores de *F-score* obtidos para cada *epoch* do modelo M7. Os valores indicados no gráfico da Figura 17, indiciam que, a partir da segunda *epoch*, os dados não melhoraram.

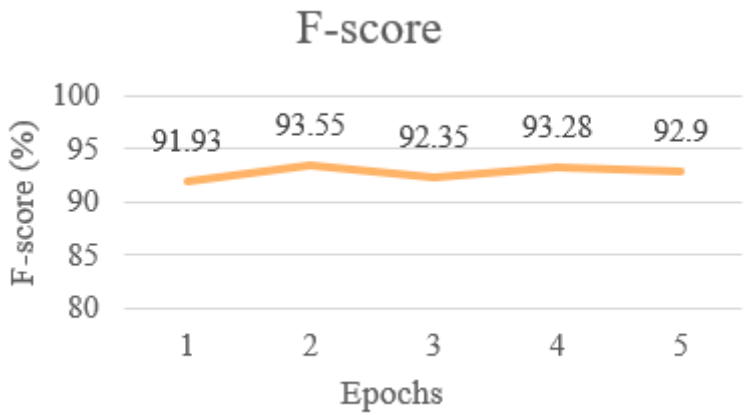

Figura 17 - Evolução do F-score do Modelo 8

Aquando da integração na plataforma doDOC, será avaliado o processamento requerido para treinar o modelo com documentos novos criados na plataforma, e a sua periodicidade, de forma a enriquecer cada vez mais os modelos existentes. Com o aumento do número de documentos na plataforma, naturalmente também aumentará o tempo de execução, no entanto, poderão ser ajustados alguns valores, nomeadamente o número de *epochs*, uma vez que o modelo estará cada vez mais enriquecido com os exemplos que irão sendo acrescentados.

4.4. Criação da Estrutura e Indexação (4ª Fase)

Após o treino do modelo a partir dos dicionários guardados no MongoDB, é possível aplicá-los ao conteúdo existente na plataforma doDOC para encontrar entidades e os seus relacionamentos. Nesta fase foi proposta uma estrutura *standard* para guardar os termos encontrados no conteúdo da plataforma. Para o tipo de conteúdo de determinado documento, será usado o respetivo modelo (de acordo com o contexto) e, as entidades e relações encontradas serão guardadas na estrutura *standard*, que permita, numa fase posterior, a execução de *queries* que retornem os resultados requerido pelo utilizador.

Esta estrutura deve permitir o armazenamento da seguinte informação: o termo encontrado (e que será depois indexado), o tipo semântico da palavra, a definição, o documento e o parágrafo onde a mesma se encontra, bem como as suas relações. As relações encontram-se divididas em duas partes, pois existem as relações encontradas no conteúdo da plataforma, e as relações que provêm do dicionário interno, criado anteriormente.

Como referido na secção anterior, para o protótipo foi treinado um modelo com dados relacionados com *clinical trials*, especificamente sobre "*depression*". Com o modelo de dados treinado, iniciou-se o processo de extração de informação de texto dos documentos da plataforma doDOC. Para isso, foram usados alguns *clinical trials* de exemplo, retirados de ClinicalTrials.gov. Estes exemplos foram importados manualmente para a plataforma doDOC.

Para a extração de informação dos documentos da plataforma doDOC, o seu conteúdo foi convertido de uma estrutura JSON para o formato de texto simples. Como referido (secção 0), os documentos da plataforma doDOC podem ser editados/redigidos através do editor que esta disponibiliza. Como existe uma grande quantidade de ações possíveis, seja formatação do texto ou outras atividades feitas pelo utilizador, como inserir citações, que precisam de ser guardadas, a estrutura que suporta e guarda o conteúdo dos documentos teve que se adaptar a essa necessidade. Para isso, a estrutura JSON assume um formato de árvore como ilustra a Figura 18, que é a representação do conteúdo guardado de um documento da plataforma.

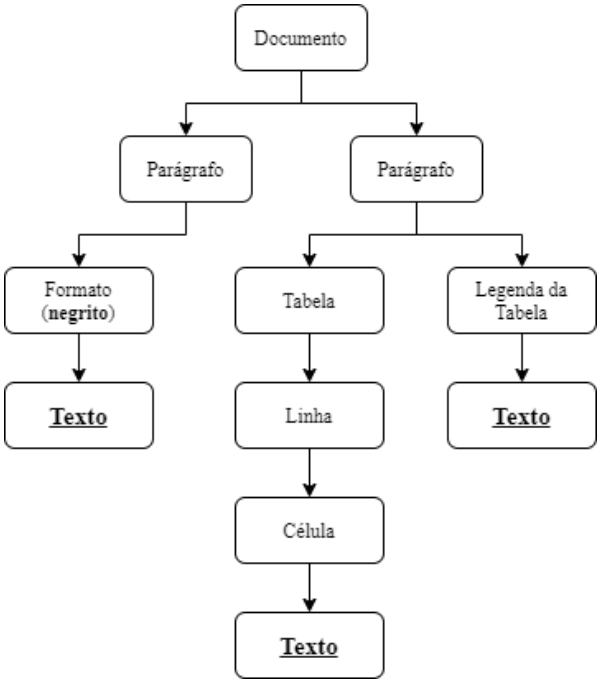

Figura 18 - Estrutura dos documentos da plataforma

Para a indexação, da estrutura ilustrada na Figura 18 só são relevantes os nós folha da árvore (os nós de texto), os restantes nós devem ser ignorados. Para o efeito implementou-se o algoritmo *Depth First Search* (DFS) para obter apenas esses nós de texto. O algoritmo DFS é um dos algoritmos de pesquisa, que usa pilhas de estruturas de dados. A capacidade deste algoritmo para encontrar um nó que ainda não foi visitado simplifica a pesquisa da solução igual nalguns problemas [58]. A ideia básica é escolher um nó inicial e empurrar todos os seus nós adjacentes para uma pilha. Retira-se um nó da pilha para selecionar o próximo nó a ser visitado e envia-se todos os seus nós adjacentes para a pilha novamente. Este processo repete-se iterativamente até que a pilha esteja vazia. Paralelamente, é necessário verificar se os nós visitados estão marcados para impedir que se visite o mesmo nó mais do que uma vez. Este algoritmo é utilizado nos documentos da doDOC para transformar as estruturas complexas de JSON em *plain text*.

Em seguida, é dada uma explicação de alguns conceitos utilizados pelo *spaCy*. Todos os termos específicos do *spaCy* para auxiliar estes conceitos estão presentes na Tabela 9. O *spaCy* contém vários módulos que facilitam o programador no processo de extração de relações. O primeiro passo é criar um `Doc`, a partir de um texto. Neste `Doc` é usado o processo de "*tokenization*" para que seja possível fazer o *parse* e o *labeling* de *tokens*. Pode ser usado o modelo "*Part-of-speech tagging*", que permite obter para um *token,* vários atributos como a *tag* que o identifica (se é um verbo, um substantivo, um símbolo, entre outros) e a dependência (*dep*) do *token* (se é um sujeito nominal, um objeto direto, etc.). Este atributo que indica a dependência permite navegar na "*Parsed Tree*". Isto é, para encontrar relações no *spaCy,* é preciso iterar numa estrutura em árvore que é criada e que permite encontrar as ligações entre os *tokens*.

Tabela 9 - Definições do spaCy

Termos	***Definição*** – spaCy
Doc	Um Doc é uma sequência de objetos *Tokens*. A partir do Doc é possível aceder a *sentences* e a *named entities*.
Entity	*Named entities* de um documento, isto é, "objetos da vida real". Ex: Nome de Países, Nome de Medicamentos
Token	Um objeto *token* individual, ou seja, uma palavra, um conjunto de palavras interligadas, símbolo de pontuação.
Pos	A *tag* simplificada do "*part-of-speech*", por exemplo: "VERB" que corresponde a verbo, "NOUN" que corresponde a substantivo, "SYM" que corresponde a símbolo.
Dep	Dependência sintática, isto é, relação entre dois *Tokens*.
Tags	*POS tags*
AUX	Auxiliar. Auxiliar de um verbo.
NOUN	Nome substantivo.
PROPN	Nome próprio.
ADP	Preposição.
VERB	Verbo.
Tag	*Dependency Tags*
Nsubj	Sujeito Nominal
Attr	Atributo
Nsubjpass	Sujeito Nominal (passivo)
Dobj	Objeto Direito
Pobj	Objeto de Preposição

Outro modelo importante no spaCy é o *Named Entity Recognition*. Uma "*named entity*" é um "objeto da vida real" [59] que está atribuída a um nome como por exemplo uma pessoa, um país, ou, neste contexto um nome de um medicamento. Estas *entities* são acedidas a partir do momento em que existe o objeto `Doc`.

Finalmente, um outro conceito utilizado pelo *spaCy*, é a capacidade de fazer *merge* de *tokens*. Isto é, o *spaCy* avalia os *tokens* e as suas *tags* e quando percebe que existem *tokens* dependentes de outros faz o *merge* dos mesmos. Isto permite que, e utilizando o exemplo da Figura 21, em vez de "*Creatine*" e "*Monohydrate*" sejam dois *tokens* separados, estes sejam unidos e seja considerado como um só *token*.

Posto isto, para proceder à extração de entidades e relações nos documentos da plataforma, foram implementadas três abordagens diferentes. Estas abordagens diferem no ponto inicial de extração, isto é, os *tokens* são todos percorridos, e, a abordagem usada é feita mediante o tipo de *token* que foi encontrado. A Figura 19 ilustra a implementação e criação de toda a estrutura.

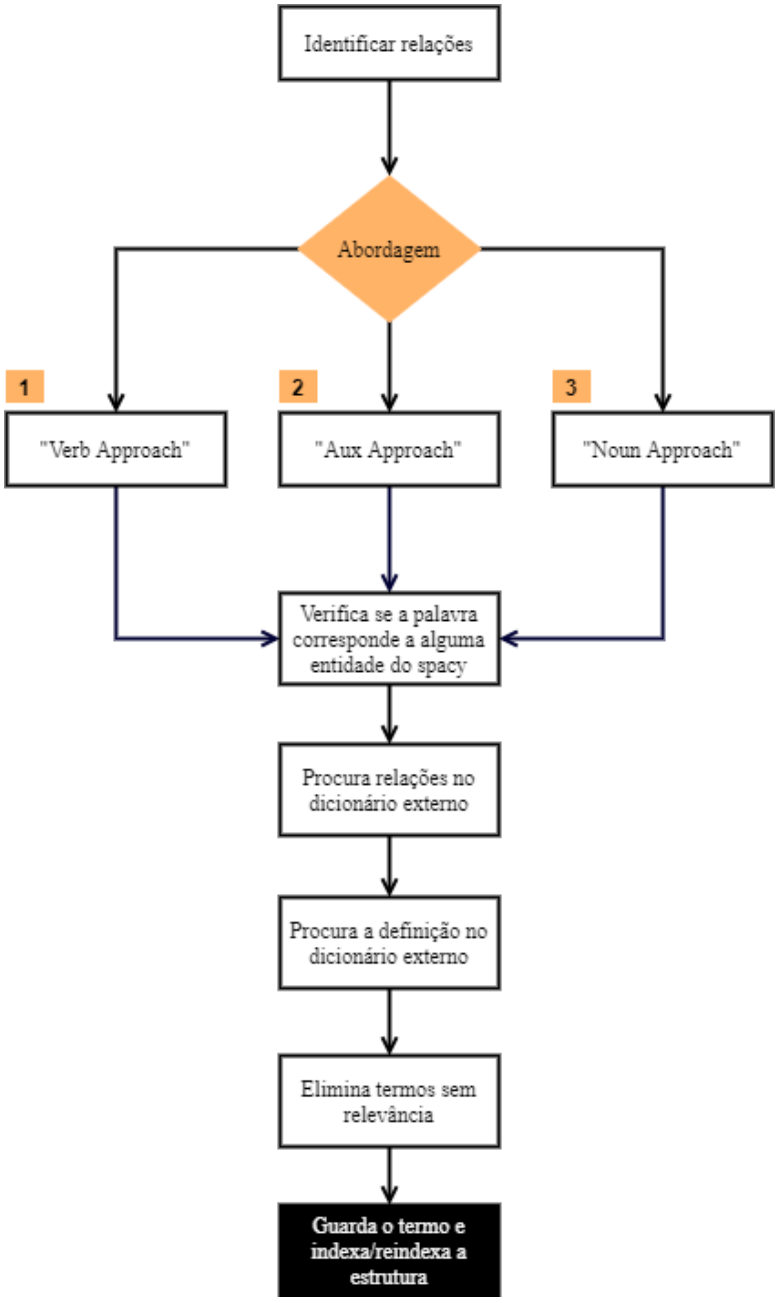

Figura 19 – Identificação de relações no spaCy

Para encontrar relações entre termos, os *tokens* são percorridos e analisados um a um, de acordo com o seu `Pos` (*tag Part-of-Speech*) e com a sua `Dep` (dependência). É de acordo com esta análise que foram usadas três abordagens (numeradas de 1 a 3 na Figura 19).

Começando pela abordagem 1: "*Verb Approach*", ilustrada na Figura 20. O primeiro passo é encontrar um verbo, neste caso, o verbo encontrado foi: "*measured*".

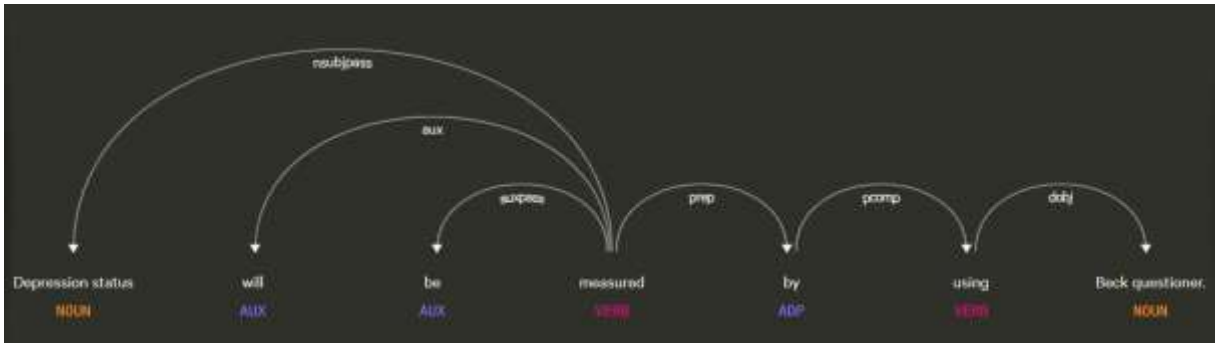

Figura 20 - *Dependency Parser* – Abordagem 1- *Verb Approach*

O segundo passo é perceber quais as dependências (representadas pelas setas) ligadas ao verbo encontrado. Neste ponto, pretende-se encontrar dois tipos de dependências, o sujeito e o objeto relacionados com o verbo. Este processo é feito percorrendo os filhos do *token* (as dependências) procurando todos os sujeitos ou as suas variantes. Neste caso será obtido o `nsubjpass` (sujeito nominal passivo): "*Depression status*". Seguidamente, procuram-se os objetos e todas as variantes dos objetos. Este passo pode não ser sempre direto, como é o caso da Figura 20, onde, para se encontrar o objeto da frase é preciso seguir um processo recursivo. Isto é, nas dependências à direita, a primeira é `prep` (modificador preposicional), e por isso não vai ser guardada. Posto isto, procuram-se os seus filhos, sucessivamente, até chegar à relação que realmente identifica o objeto, neste caso, `dobj` (objeto direto): "*Black questioner*".

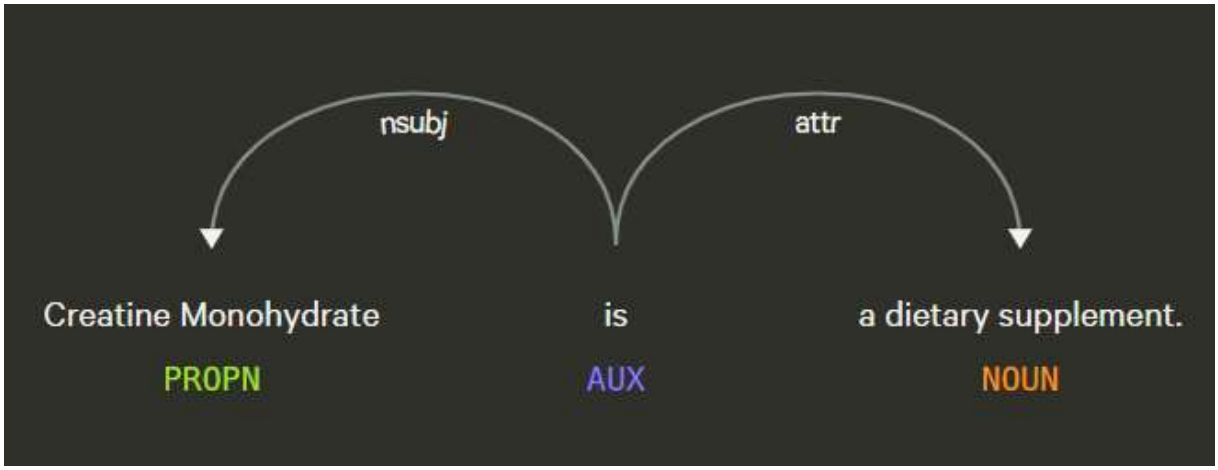

Figura 21 - *Dependency Parser* - Abordagem 2 - *Aux Approach*

A abordagem 2 é a "*Aux Approach*", ilustrada Figura 21. Para extrair a relação o *token* que se procura é o `AUX` (auxiliar). Neste caso, a partir das dependências é possível extrair o sujeito: "*Creatine Monohydrate*". Especificamente nesta abordagem, por ser um *token* auxiliar, na ausência de se encontrarem dependências de objetos, procuram-se também atributos. O atributo encontrado foi: "*a dietary supplement*".

Por fim, na abordagem 3, "*Noun Approach*", ilustrada na Figura 22, o objetivo é encontrar um substantivo (`NOUN`) e, a partir daí, perceber se tem como dependência o verbo e o objeto com o qual o substantivo se relaciona. Neste caso, o *token* identificado como `NOUN` (substantivo), é: "*an FDA approved food additive*". Navegando pelos seus "filhos", é encontrado o `VERB`

(verbo): "*made*". Finalmente, e recorrendo mais uma vez a uma pesquisa recursiva, encontra-se a dependência `pobj` (objeto de preposição): "*A highly processed starch product*".

Figura 22 - *Dependency Parser* - Abordagem 3 - *Noun Approach*

Com as relações encontradas, passou-se ao passo seguinte, que é perceber se o termo encontrado, corresponde a alguma entidade treinada no modelo. Por exemplo, no caso de "*Creatine Monohydrate*", como é uma entidade que foi anteriormente treinada no modelo, esta é identificada com uma entidade em que o seu tipo semântico é: "*Amino Acid, Peptide, or Protein*".

Após estes passos, procura-se o termo no dicionário criado, por forma a encontrar as relações que possam existir e a definição do mesmo.

O termo, por fim, é indexado como texto (uma das funcionalidades do *MongoDB*), ficando apto para fazer pesquisas de texto (*Full Text Search*). O JSON abaixo representa um exemplo de uma entrada da estrutura criada. No Apêndice C, encontra-se o JSON *schema*, com a explicação desta estrutura.

```
{
    "_id"        : ObjectId("5e768d6f8035730cbeebcf41"),
    "definition" :  "An amino acid that occurs in vertebrate tissues
                     and urine. In muscle tissue, creatine generally
                     occurs as phosphocreatine. Creatine is excreted as
                     CREATININE in the urine.",
    "dicts"      : {
           "metathesaurus_subset" : [
            {
                "relation" : "hierarchy",
                "name" : "phosphoserine"
            }
        ]
    },
    "location"  : "ddc8c1213a2-1689-4cdf-8ae6-24339964f723",
    "document"  : "5dc96765803573106847c900",
    "semantic_type" : "Amino Acid, Peptide, or Protein",
    "word"      : "creatine monohydrate",
    "relations" : [
        {
            "sentence" : "The purpose of this study is to see if
                          creatine monohydrate is effective in
                          preventing altitude-induced depression.",
            "object"   : "altitude-induced depression",
            "relation" : "preventing"
        }
    ]
}
```

Este exemplo corresponde ao termo encontrado "*creatine monohydrate*" (representado no exemplo por *word*). As relações encontradas no conteúdo dos documentos da plataforma doDOC estão representadas por "*relations*", e permitem concluir que o termo: "*creatine monohydrate*" tem uma relação "*preventing*" com o termo "*altitude-induced depression*". No caso das relações que provêm do dicionário interno criado, é possível, através do exemplo, perceber que o dicionário onde foram encontradas foi "*metathesaurus_subset*", dicionário criado (secção 4.2), e que há uma relação de hierarquia com o termo "*phosphoserine*". O exemplo também inclui a definição do termo, bem como a sua localização no documento e a frase onde se encontra.

4.5. Desenvolvimento do *Endpoint* (5ª Fase)

Com a estrutura de dados definida e com o conteúdo indexado com atualizações do utilizador, estão reunidas todas as condições para o desenvolvimento do *endpoint*, que permite responder às pesquisas do utilizador.

Na pesquisa propriamente dita, as funcionalidades de pesquisa de texto do *MongoDB* funciona de acordo com as expectativas e objetivos traçados. As funcionalidades de *Full Text Search* do *MongoDB,* estão explicadas na secção 3.4.

O objetivo deste *endpoint* é devolver todas as informações consideradas necessárias para a pesquisa do utilizador. Para um melhor entendimento da abordagem seguida na apresentação dos resultados, foram definidas algumas palavras-chave:

- **`doDOC id`**: identificador do documento na plataforma doDOC onde a pesquisa está a ser efetuada;
- **`Document id`**: Cada pesquisa efetuada poderá retornar um ou mais resultados, cada resultado tem associado o identificador do documento na plataforma doDOC (`doDOC id`).

Posto isto, após a pesquisa no *MongoDB* e de este retornar todos os resultados possíveis, seguiu-se a abordagem ilustrada na Figura 23.

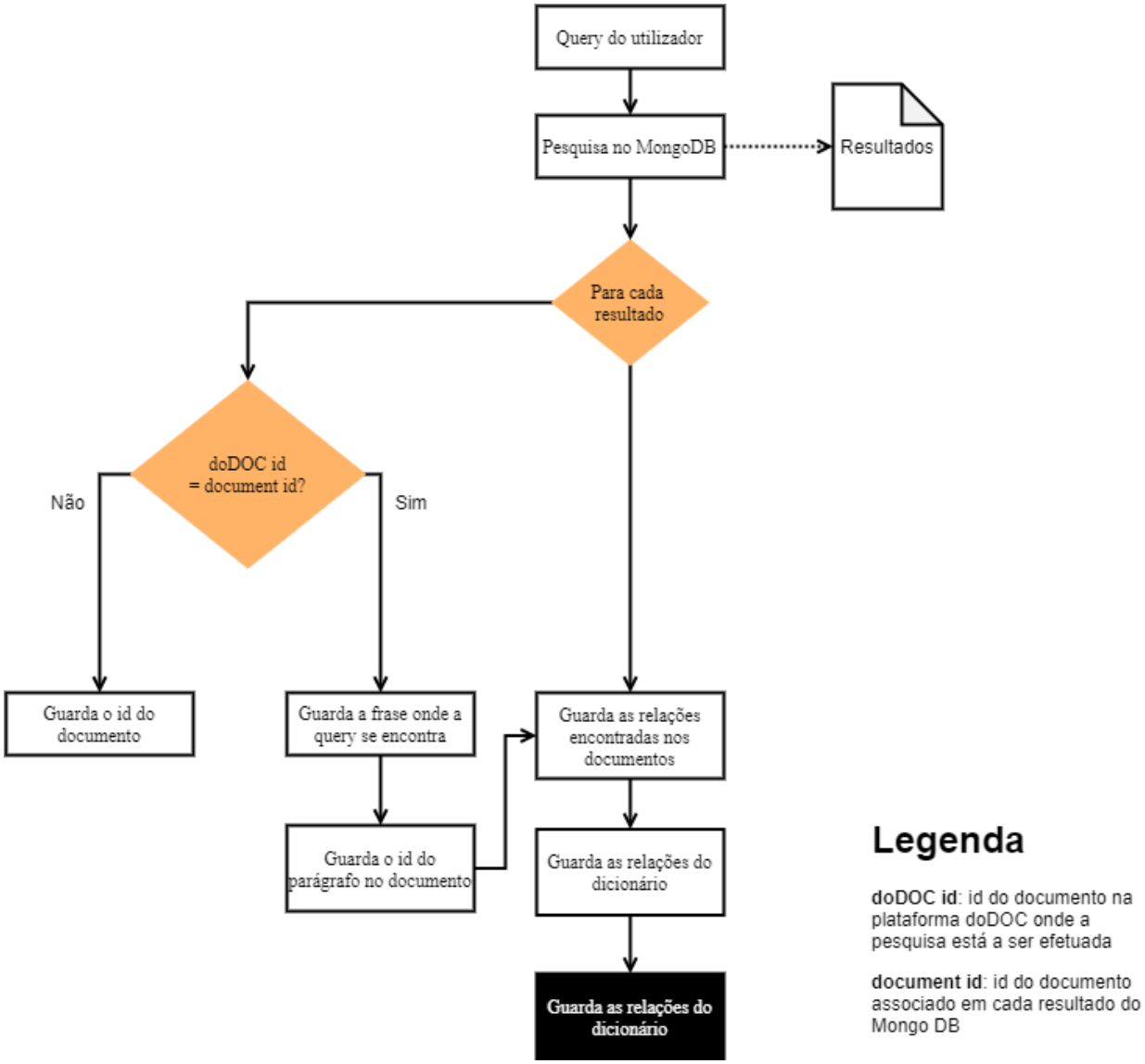

Figura 23 - Abordagem usada no Endpoint

O *endpoint* está preparado para receber dois parâmetros: a *query* do utilizador (palavra que ele pretende pesquisar) e o `doDOC id` (id do documento na plataforma). A *query* do utilizador é utilizada para fazer uma pesquisa para encontrar os documentos do MongoDB que a contêm. Esta pesquisa é feita recorrendo à funcionalidade do MongoDB de "*text search*". Esta pesquisa poderá retornar um ou mais resultados.

Para cada resultado, o `doDOC id` (id do documento na plataforma) é comparado ao `document id` (id do documento na plataforma presente no resultado). Se os ids forem diferentes, esse id é guardado e não são guardadas as frases que possuem a *query* (termo pesquisado pelo utilizador). Se, por outro lado, os ids forem iguais, a frase onde a *query* se encontra é guardada, juntamente com o id do parágrafo para que seja possível fazer uma navegação para o mesmo.

As relações associadas ao conteúdo e as relações do dicionário serão guardadas, independentemente do documento da plataforma doDOC onde se encontram. Toda a informação guardada é retornada no *endpoint* para ser apresentada ao utilizador.

A Figura 25 ilustra ao ecrã do protótipo feito pela equipa de design da doDOC e reflete o resultado obtido através da ferramenta Postman[4], ilustrado na Figura 24.

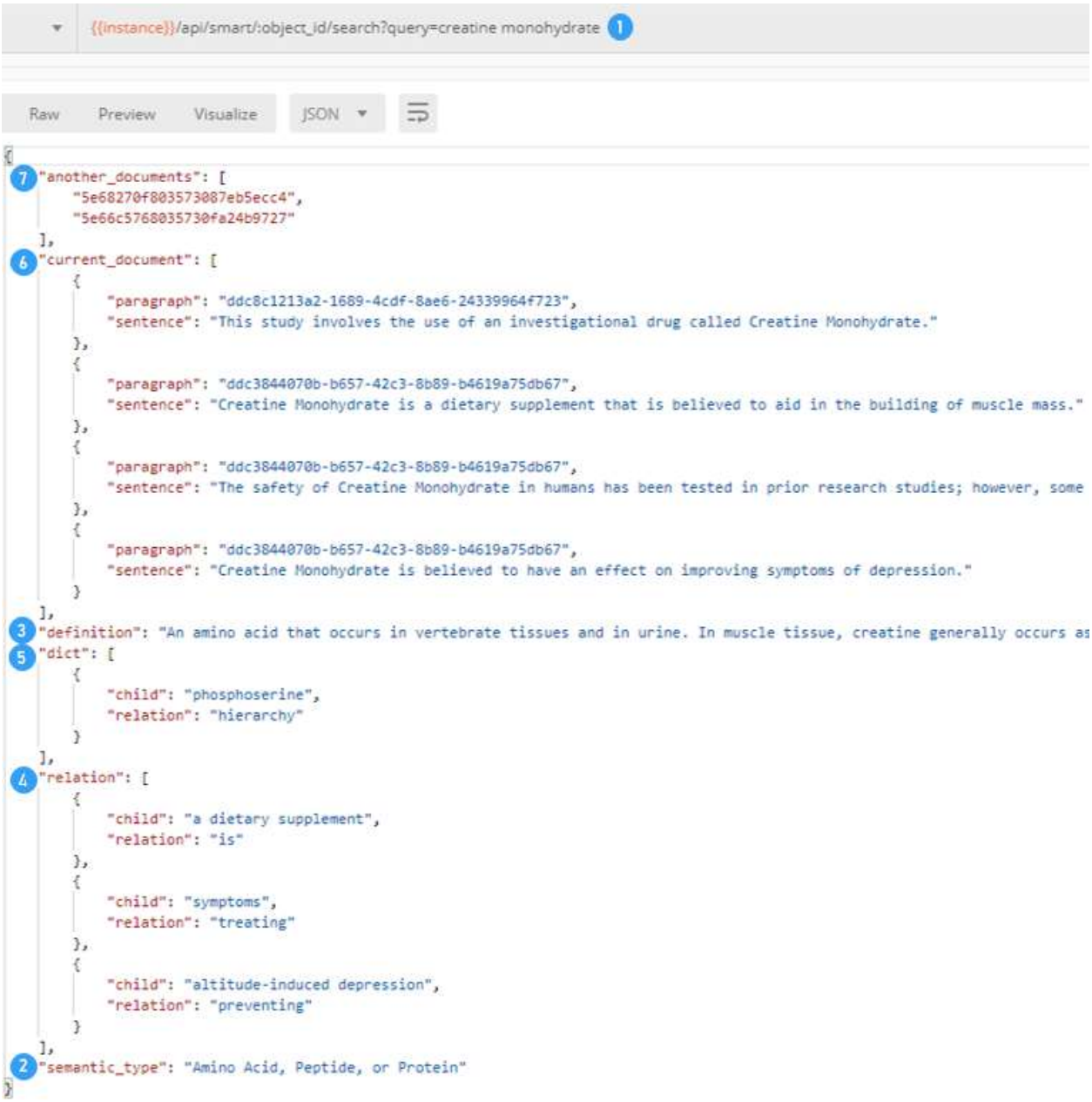

Figura 24 - Demonstração da Resposta do Endpoint através do Postman

[4] Postman é uma ferramenta que tem como objetivo testar serviços RESTful (Web APIs).

As duas figuras permitem uma melhor perceção da abordagem seguida. O *endpoint* recebe no parâmetro *query* o valor "*creatine monohydrate*", como se pode visualizar no ponto 1 da Figura 24 e da Figura 25. A informação obtida pelo utilizador é a indicada do ponto 2 ao ponto 7 (em ambas as figuras). Por exemplo, é possível perceber que o tipo semântico da *query*, (como representa o ponto 2) é "*Amino Acid, peptide or protein*". O ponto 3, por sua vez, apresenta ao utilizador a definição do termo, o 4 e 5 as suas relações, provenientes do conteúdo dos documentos da plataforma e do dicionário criado na secção 4.2, respetivamente. Neste ponto é possível verificar que, a relação "*is*" -> "*a dietary supplement*" é a relação encontrada através da abordagem *"aux"* ilustrada na Figura 21 da secção 4.4. O ponto 6 apresenta as frases encontradas no documento aberto como se pode ver no *Mockup* (Figura 25), e, finalmente, o ponto 7 representa os outros documentos encontrados na plataforma que contêm a *query* pesquisada pelo utilizador. A pesquisa retornada ao utilizador tem um tempo de execução de 23 milisegundos.

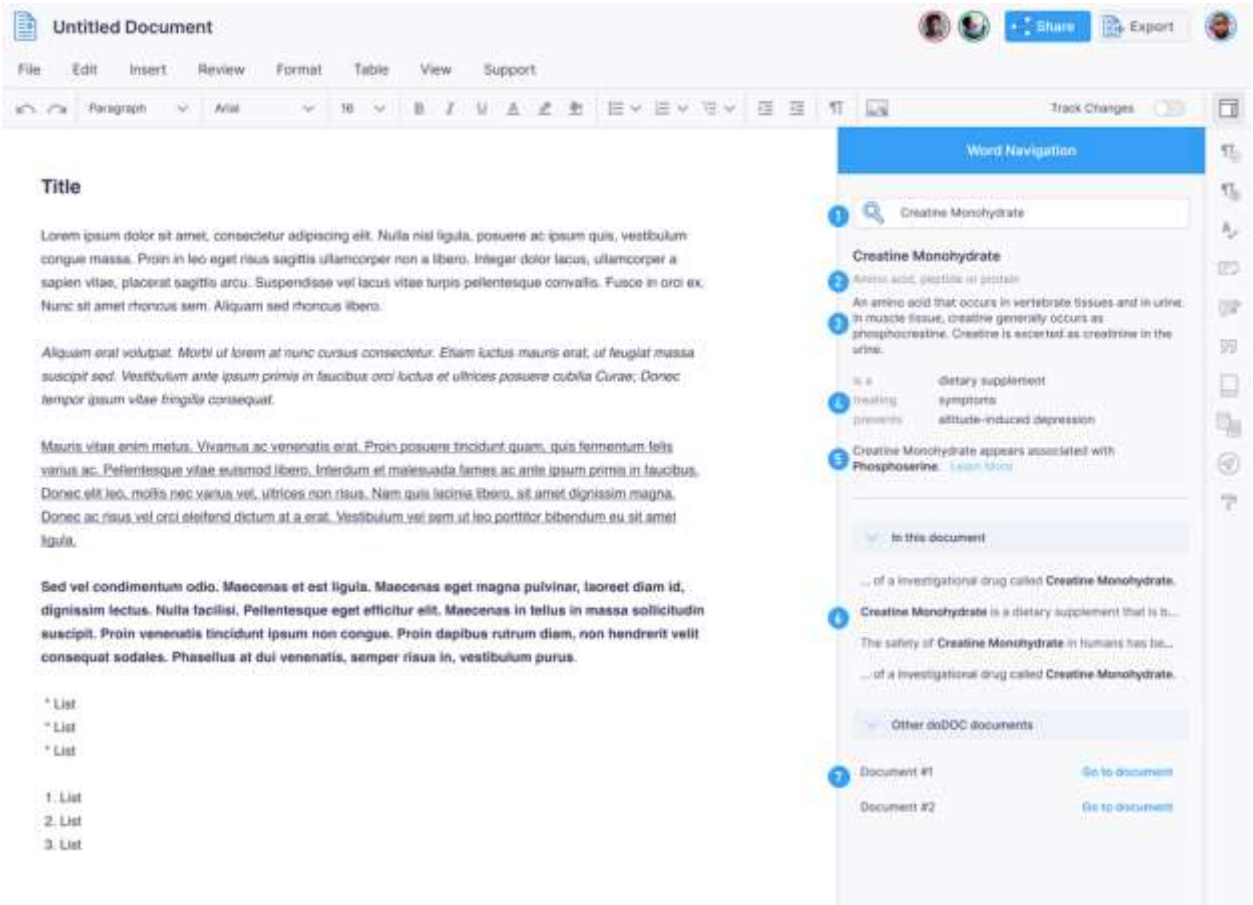

Figura 25 - *Mockup* do Protótipo

Existem várias abordagens para a apresentação, ao utilizador, das relações encontradas, todavia, necessitam de uma avaliação feita pelos utilizadores finais da plataforma. Numa primeira fase, a abordagem implementada foca-se na distância entre palavras numa relação, sendo que se consideram mais relevantes, e, portanto, têm precedência na apresentação ao utilizador, as relações que possuem menos palavras entre elas.

5. Conclusões

Este trabalho teve como objetivo identificar potenciais abordagens e propor uma estratégia de extração de informação que seja integrável com uma solução de escrita colaborativa e assim permita agilizar tanto o processo de escrita como de verificação e validação de documentos.

Algumas das ferramentas encontradas são muito completas para a extração de informação, no entanto é necessário existir um equilíbrio entre as potencialidades das ferramentas existentes e a facilidade de integração num sistema robusto como é a plataforma doDOC. O objetivo da abordagem proposta é que seja integrável numa aplicação já em fase de desenvolvimento avançado. Com esta consideração, existiram decisões que foram tomadas mediante a arquitetura já existente.

O trabalho envolveu uma análise comparativa de ferramentas usadas para extração de informação, e ferramentas de extração de informação usadas para o contexto de *clinical trials.* Seguidamente, foram filtradas as ferramentas que se adequavam ao desenvolvimento do protótipo. Foi também feita uma análise de ferramentas usadas para indexação e *full text search*. Foram identificadas algumas fontes médicas relevantes para o contexto de *clinical trials* e realizado um estudo do repositório de informação *Metathesaurus* com o intuito de o integrar no protótipo. Foi proposta uma estrutura *standard* para a incorporação desta informação na construção de um dicionário com termos médicos.

Este estudo resultou na extração e processamento das terminologias e descreve como podem ser usadas para o treino de modelos *multi-task* CNN. Para isso, foi feita uma análise da ferramenta *spaCy* e de casos de uso com a utilização da mesma. Após o treino e avaliação de vários modelos de acordo com a variação de parâmetros, identificou-se o modelo que apresenta com melhores resultados e que foi aplicado para extrair informação. De modo a aumentar a quantidade de frases e relações contempladas na extração, foram analisadas e implementadas três abordagens diferentes: "*verb approach*", "*aux approach*" e "*noun approach*". Posteriormente, foi proposta uma estrutura *standard* com os termos extraídos, para posterior indexação. Para a integração com a plataforma, foi posteriormente definido e desenvolvido um *endpoint* para receber uma *query* do utilizador e retornar os resultados da pesquisa feita na estrutura indexada.

Ao longo deste trabalho, existiram algumas dificuldades, nomeadamente na área da extração de informação, existindo a necessidade de limitar o âmbito para o contexto: "*depression*", e no treino do modelo e na determinação da configuração dos parâmetros que permitem a obter melhores resultados.

O protótipo ainda está em fase de teste, sendo que ainda não foi implementado em ambiente de produção com a plataforma doDOC. No entanto, a funcionalidade *full text search* foi isolada

do restante protótipo e já se encontra integrada com a plataforma. Com esta integração foi possível perceber que uma indexação ativada por cada alteração do utilizador no documento causa problemas de *performance* na plataforma.

Para trabalho futuro, ficam algumas funcionalidades tais como a extensão do protótipo para outras línguas; a melhoria contínua de extração de relações, adicionando outras abordagens para diferentes casos, de forma a contemplar mais frases e mais relações. É necessária a avaliação da performance em ambiente de produção da extração: uma vez que esta será feita em *background* pode não atrapalhar a *performance* da plataforma, mas quando escalada poderá trazer alguns problemas.

Poderá ser interessante avaliar a possibilidade, no treino do modelo, da utilização da variância do valor do *f-score* da avaliação modelo na determinação do número de *epochs* de treino. Isto é, se for possível comprovar que, a partir de um determinado número de *epochs*, a variância entre os valores de *f-score* para um determinado modelo é cada vez menor, então, é possível usar o valor dessa variância para determinar o número de *epochs* necessárias para treinar o modelo. Sendo assim, esse valor deixaria de ser estático. Para além da variância, poderá ser tido em conta, o tempo de execução, permitindo que, mesmo que exista um grande volume de dados (sendo que a tendência natural é que o número de documentos da plataforma doDOC aumente cada vez mais), o treino possa ser ajustado.

A pertinência dos resultados apresentados poderá ser refinado com a utilização do feedback do utilizador no resultado das pesquisas e integração dessa avaliação no próprio modelo de treino, para isso, poderá ser feito um sistema de avaliação para as relações retornadas ao utilizador. Isto é, uma otimização daa abordagem implementada e descrita na secção 4.5, em que a pertinência dos resultados é relacionado com a distância entre palavras, ou seja, apresenta-se com precedência ao utilizador as relações que possuem menos palavras entre elas. Para uma futura investigação, será implementado um método que contabilize o número de vezes que a relação foi escolhida, sendo que as relações que o utilizador mais teve interesse em explorar, serão as primeiras a serem apresentadas. Outra proposta possível, será atribuir graus de confiança às diferentes abordagens utilizadas para fazer a extração de relações. Isto é, avaliar o número e a qualidade de relações encontradas por cada abordagem e atribuir um peso maior às abordagens com melhores resultados.

Considero que os resultados foram bastante positivos, uma vez que o protótipo se apresenta completamente funcional e apresenta resultados coerentes. Existe ainda algumas sugestões que seriam interessantes implementar, mas, o objetivo de ajudar investigadores e escritores na área da medicina, na sua produtividade, estará mais perto e mais possível com este estudo.

Referências

[1] "The Seven Practice Areas of Text Analytics," in *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*, Elsevier, 2012, pp. 29–41.

[2] "What is Information Extraction?," *Ontotext*. https://www.ontotext.com/knowledgehub/fundamentals/information-extraction/ (accessed Jan. 19, 2020).

[3] M. Elsadig, A. Ahmed, and M. Himmat, "Information Extraction methods and extraction techniques in the chemical document's contents: Survey," *ARPN J. Eng. Appl. Sci.*, vol. 10, pp. 1068–1073, Feb. 2015.

[4] C. Niklaus, M. Cetto, A. Freitas, and S. Handschuh, "A Survey on Open Information Extraction," in *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA, Aug. 2018, pp. 3866–3878, Accessed: Mar. 10, 2020. [Online]. Available: https://www.aclweb.org/anthology/C18-1326.

[5] C. Liu, W. Sun, W. Chao, and W. Che, "Convolution Neural Network for Relation Extraction," in *Advanced Data Mining and Applications*, Berlin, Heidelberg, 2013, pp. 231–242, doi: 10.1007/978-3-642-53917-6_21.

[6] D. Wimalasuriya and D. Dou, "Ontology-based information extraction: An Introduction and a survey of current approaches," *J Inf. Sci.*, vol. 36, pp. 306–323, May 2010, doi: 10.1177/0165551509360123.

[7] St. consortium, "RUTI vaccine proceeds through Phase IIb Clinical Trials," *STriTuVaD*, Jan. 22, 2020. https://www.strituvad.eu/post/ruti-vaccine-proceeds-through-phase-iib-clinical-trials (accessed Apr. 23, 2020).

[8] Mausam, M. Schmitz, S. Soderland, R. Bart, and O. Etzioni, "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*, Jeju Island, Korea, Jul. 2012, pp. 523–534, Accessed: Jan. 01, 2019. [Online]. Available: http://www.aclweb.org/anthology/D12-1048.

[9] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in *Proceedings of the 20th international joint conference on Artifical intelligence*, Hyderabad, India, Jan. 2007, pp. 2670–2676, Accessed: Mar. 10, 2020. [Online].

[10] F. Wu and D. S. Weld, "Open Information Extraction Using Wikipedia," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, Jul. 2010, pp. 118–127, Accessed: Mar. 10, 2020. [Online]. Available: https://www.aclweb.org/anthology/P10-1013.

[11] A. Fader, S. Soderland, and O. Etzioni, "Identifying Relations for Open Information Extraction," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, UK., Jul. 2011, pp. 1535–1545, Accessed: Mar. 10, 2020. [Online]. Available: https://www.aclweb.org/anthology/D11-1142.

[12] A. Akbik and A. Löser, "KrakeN: N-ary Facts in Open Information Extraction," in *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, Montrèal, Canada, 2012, pp. 52–56, Accessed: Jan. 01, 2019. [Online]. Available: http://aclweb.org/anthology/W12-3010.

[13] J. Schmidek and D. Barbosa, "Improving Open Relation Extraction via Sentence Re-Structuring," May 2014.

[14] N. Bhutani, H. V. Jagadish, and D. Radev, "Nested Propositions in Open Information Extraction," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, Nov. 2016, pp. 55–64, Accessed: Jan. 01, 2019. [Online]. Available: https://aclweb.org/anthology/D16-1006.

[15] "Universal Dependencies." https://universaldependencies.org/guidelines.html (accessed Jan. 19, 2020).
[16] M. Cetto, C. Niklaus, A. Freitas, and S. Handschuh, "Graphene: Semantically-Linked Propositions in Open Information Extraction," in *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA, Aug. 2018, pp. 2300–2311, Accessed: Mar. 10, 2020. [Online]. Available: https://www.aclweb.org/anthology/C18-1195.
[17] P.-Y. Chen, Y.-H. Lee, Y.-H. Wu, and W.-Y. Ma, "IExM: Information Extraction System for Movies," Apr. 2017, pp. 189–193, doi: 10.1145/3041021.3054729.
[18] P. N. Golshan, M. Student, and L. Safari, "A Study of Recent Contributions on Information Extraction," p. 6.
[19] P. Li and H. Huang, "Clinical Information Extraction via Convolutional Neural Network," *ArXiv160309381 Cs*, Mar. 2016, Accessed: Dec. 02, 2018. [Online]. Available: http://arxiv.org/abs/1603.09381.
[20] F. Gutierrez, D. Dou, S. Fickas, D. Wimalasuriya, and H. Zong, "A hybrid ontology-based information extraction system," *J. Inf. Sci.*, vol. 42, no. 6, pp. 798–820, Dec. 2016, doi: 10.1177/0165551515610989.
[21] Y. Wang *et al.*, "Clinical information extraction applications: A literature review," *J. Biomed. Inform.*, vol. 77, pp. 34–49, Jan. 2018, doi: 10.1016/j.jbi.2017.11.011.
[22] D. Ferrucci and A. Lally, "UIMA: an architectural approach to unstructured information processing in the corporate research environment," *Nat. Lang. Eng.*, vol. 10, no. 3–4, pp. 327–348, Sep. 2004, doi: 10.1017/S1351324904003523.
[23] A. R. Aronson and F.-M. Lang, "An overview of MetaMap: historical perspective and recent advances," *J. Am. Med. Inform. Assoc. JAMIA*, vol. 17, no. 3, pp. 229–236, Jun. 2010, doi: 10.1136/jamia.2009.002733.
[24] "Apache OpenNLP." https://opennlp.apache.org/ (accessed Mar. 10, 2020).
[25] J. C. Denny, P. R. Irani, F. H. Wehbe, J. D. Smithers, and A. Spickard, "The KnowledgeMap Project: Development of a Concept-Based Medical School Curriculum Database," *AMIA. Annu. Symp. Proc.*, vol. 2003, pp. 195–199, 2003.
[26] S. Goryachev, M. Sordo, and Q. T. Zeng, "A Suite of Natural Language Processing Tools Developed for the I2B2 Project," *AMIA. Annu. Symp. Proc.*, vol. 2006, p. 931, 2006.
[27] J. Cowie and Y. Wilks, *Information Extraction*. 1996.
[28] A. R. Aronson, J. G. Mork, A. Névéol, S. E. Shooshan, and D. Demner-Fushman, "Methodology for Creating UMLS Content Views Appropriate for Biomedical Natural Language Processing," *AMIA. Annu. Symp. Proc.*, vol. 2008, pp. 21–25, 2008.
[29] C. Friedman, G. Hripcsak, W. DuMouchel, S. B. Johnson, and P. D. Clayton, "Natural language processing in an operational clinical information system," *Nat. Lang. Eng.*, vol. 1, no. 1, pp. 83–108, Mar. 1995, doi: 10.1017/S1351324900000061.
[30] S. Sohn, C. Clark, S. R. Halgrim, S. P. Murphy, C. G. Chute, and H. Liu, "MedXN: an open source medication extraction and normalization tool for clinical text," *J. Am. Med. Inform. Assoc. JAMIA*, vol. 21, no. 5, pp. 858–865, Oct. 2014, doi: 10.1136/amiajnl-2013-002190.
[31] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, "GATE: an Architecture for Development of Robust HLT applications," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Pennsylvania, USA, Jul. 2002, pp. 168–175, doi: 10.3115/1073083.1073112.
[32] M. A. Musen, "The Protégé Project: A Look Back and a Look Forward," *AI Matters*, vol. 1, no. 4, pp. 4–12, Jun. 2015, doi: 10.1145/2757001.2757003.
[33] C. Friedman, L. Shagina, S. A. Socratous, and X. Zeng, "A WEB-Based Version of MedLEE: A Medical Language Extraction and Encoding System," *Proc. AMIA Annu. Fall Symp.*, p. 938, 1996.
[34] C. Zheng, N. Rashid, R. Koblick, and J. An, "Medication Extraction from Electronic Clinical Notes in an Integrated Health System: A Study on Aspirin Use in Patients with Nonvalvular Atrial Fibrillation," *Clin. Ther.*, vol. 37, no. 9, pp. 2048-2058.e2, Sep. 2015, doi: 10.1016/j.clinthera.2015.07.002.
[35] "NLP4J by emorynlp." https://emorynlp.github.io/nlp4j/ (accessed Mar. 25, 2020).

[36] "The MATE workbench – An annotation tool for XML coded speech corpora - ScienceDirect." https://www.sciencedirect.com/science/article/abs/pii/S0167639300000716 (accessed Mar. 25, 2020).
[37] "Facts & Figures · spaCy Usage Documentation." https://spacy.io/usage/facts-figures (accessed Jan. 29, 2020).
[38] J. D. Choi, J. Tetreault, and A. Stent, "It Depends: Dependency Parser Comparison Using A Web-based Evaluation Tool," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Beijing, China, Jul. 2015, pp. 387–396, doi: 10.3115/v1/P15-1038.
[39] *allenai/scispacy*. AI2, 2020.
[40] L. Soldaini, "QuickUMLS : a fast , unsupervised approach for medical concept extraction," 2016.
[41] "UMLS Metathesaurus - CPT (CPT - Current Procedural Terminology) - Synopsis." https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/CPT/index.html (accessed Feb. 01, 2020).
[42] "UMLS Metathesaurus - ICD10CM (International Classification of Diseases, Tenth Revision, Clinical Modification) - Synopsis." https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/ICD10CM/index.html (accessed Feb. 01, 2020).
[43] "UMLS Metathesaurus - LNC (LOINC) - Synopsis." https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/LNC/index.html (accessed Feb. 01, 2020).
[44] "UMLS Metathesaurus - MSH (MeSH) - Synopsis." https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/MSH/index.html (accessed Feb. 01, 2020).
[45] "UMLS Metathesaurus - RXNORM (RXNORM) - Synopsis." https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/RXNORM/index.html (accessed Feb. 01, 2020).
[46] "UMLS Metathesaurus - SNM (SNOMED 1982) - Synopsis." https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/SNM/index.html (accessed Feb. 01, 2020).
[47] "Unified Medical Language System (UMLS)." https://www.nlm.nih.gov/research/umls/index.html (accessed Jan. 24, 2020).
[48] S. B. Online, "Chapter 4 MongoDB Functionality - Data Modeling for MongoDB." https://learning.oreilly.com/library/view/data-modeling-for/9781935504702/Chapters-7.xhtml (accessed Jan. 10, 2020).
[49] "Using Python | Apache Solr Reference Guide 6.6." https://lucene.apache.org/solr/guide/6_6/using-python.html (accessed Jan. 24, 2020).
[50] G. P. Purcell and D. D. Mar, "SCOUT: information retrieval from full-text medical literature.," *Proc. Annu. Symp. Comput. Appl. Med. Care*, pp. 91–95, 1992.
[51] K. Chodorow, *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*. O'Reilly Media, Inc., 2013.
[52] "Text Search — MongoDB Manual," *https://github.com/mongodb/docs/blob/v4.2/source/text-search.txt*. https://docs.mongodb.com/manual/text-search (accessed Feb. 02, 2020).
[53] N. C. for B. Information, U. S. N. L. of M. 8600 R. Pike, B. MD, and 20894 Usa, *Metathesaurus*. National Library of Medicine (US), 2009.
[54] "Metathesaurus Data Files." https://www.nlm.nih.gov/research/umls/new_users/online_learning/Meta_006.html (accessed Feb. 01, 2020).
[55] "English · spaCy Models Documentation," *English*. https://spacy.io/models/en (accessed Jan. 29, 2020).
[56] S. SHARMA, "Epoch vs Batch Size vs Iterations," *Medium*, Mar. 05, 2019. https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9 (accessed Apr. 30, 2020).

[57] W. Sun, Z. Cai, Y. Li, F. Liu, S. Fang, and G. Wang, “Data Processing and Text Mining Technologies on Electronic Medical Records: A Review,” *Journal of Healthcare Engineering*, 2018. https://www.hindawi.com/journals/jhe/2018/4302425/ (accessed Apr. 26, 2020).
[58] “(PDF) Implementation and Analysis of Depth-First Search (DFS) Algorithm for Finding The Longest Path,” *ResearchGate*. https://www.researchgate.net/publication/264286830_Implementation_and_Analysis_of_Depth-First_Search_DFS_Algorithm_for_Finding_The_Longest_Path (accessed Apr. 26, 2020).
[59] “Linguistic Features · spaCy Usage Documentation.” https://spacy.io/usage/linguistic-features (accessed Mar. 29, 2020).

Apêndice A – Descrição dos Atributos (Metathesaurus)

Atributo	*Descrição*	*Ficheiros*
RELA	Descreve a relação entre conceitos	MRAUI.RRF MRCUI.RRF MRSMAP.RRF MRHIER.RRF MRREL.RRF MRMAP.RRF
ATN	Nome do Atributo	MRSAT.RRF
DEF	Definição do conceito	MRDEF.RRF
EXPL	Explicação detalhada	MRDOC.RRF
LAT	Termo da Linguagem	MRCONSO.RRF
NSTR	String normalizada	MRXNS_ENG.RRF
NWD	Palavra normalizada	MRXNW_ENG.RRF
REL	Relação	MRAUI.RRF MRMAP.RRF MRSMAP.RRF MRCUI.RRF MRREL.RRF
STY	Tipo de Semântica	MRSTY.RRF
STR	Nome de conceito	MRCONSO.RRF
STT	Tipo de *string*	MRCONSO.RRF
TTY	Tipo do termo na fonte	MRCONSO.RRF MRRANK.RRF
TYPE	Tipo de informação	MRDOC.RRF
VALUE	Valor	MRRDOC.RRF

Apêndice B – JSON Schema (Metathesaurus Subset)

```
{
  "$schema": "http://json-schema.org/draft-07/schema",
  "$id": "http://example.com/example.json",
  "type": "array",
  "title": "Metathesaurus Subset",
  "description": "Dicionário do contexto clinical trials",
  "items": {
    "$id": "#/items",
    "title": "Exemplo",
    "description": "Exemplo de uma entrada no dicionário criado",
    "required": ["_id", "language", "name", "semantic_type"],
    "properties": {
      "id": {
        "type": "ObjectId",
        "description": "Identificador do MongoDB para esta entrada"
      },
      "source_id": {
        "type": "string",
        "description": "Identificador do termo usado pela sua fonte"
      },
      "language": {
        "type": "string",
        "description": "Língua em que o termo se encontra"
      },
      "name": {
        "type": "string",
        "description": "Nome do termo"
      },
      "semantic_type": {
        "type": "string",
        "description": "Tipo semântico do termo, de acordo com fonte"
      },
      "definition": {
        "type": "string",
        "description": "A definição do termo, de acordo com a fonte"
      },
      "relations": {
        "type": "array",
        "description": "Lista de relações nas coleções MRREL MRHIER",
        "items": {
          "type": "object",
          "description": "Exemplo de uma entrada da lista",
          "required": ["source", "child_name"],
          "properties": {
            "relation": {
              "type": "string",
              "description": "A relação entre os dois termos."
            },
            "child_id": {
              "type": "string",
              "description": "Identificador do termo relacionado da fonte"
            },
            "source": {
              "type": "string",
              "description": "Origem da relação (coleção onde foi encontrada)"
            },
            "child_name": {
              "type": "string",
              "description": "O nome do termo relacionado"
            }
          }
        }
      }
    }
```

```
    }
}
```

B2

Apêndice C – JSON Schema (Estrutura Final)

```
{
  "$schema": "http://json-schema.org/draft-07/schema",
  "$id": "http://example.com/example.json",
  "type": "array",
  "title": "Estrutura Final",
  "description": "Estrutura com termos indexados, para pesquisa do
                  utilizador",
  "items": {
    "type": "object",
    "description": "Exemplo de uma das entradas na estrutura",
    "required": ["_id", "location", "document", "semantic_type", "word"],
    "properties": {
      "_id": {
        "type": "ObjectId",
        "description": "Identificador do MongoDB para esta entrada"
      },
      "definition": {
        "type": "string",
        "description": "Definição do termo, do dicionário Metathesaurus
                        Subset"
      },
      "location": {
        "type": "string",
        "description": "Identificador do parágrafo onde a palavra se encontra"
      },
      "document": {
        "type": "string",
        "description": "Identificador do documento onde a palavra se encontra"
      },
      "semantic_type": {
        "type": "string",
        "description": "Tipo semântico, d o dicionário Metathesaurus Subset"
      },
      "word": {
        "type": "string",
        "description": "O nome do termo, que será indexado"
      },
      "dicts": {
        "type": "object",
        "description": "Relações provenientes dos dicionários",
        "properties": {
          "metathesaurus_subset": {
            "type": "array",
            "description": "Dicionário onde se encontra a relação",
            "items": {
              "type": "object",
              "description": "Relação encontrada com o termo do campo word",
              "required": ["name"],
              "properties": {
                "relation": {
                  "type": "string",
                  "description": "Nome da relação"
                },
                "name": {
                  "type": "string",
                  "description": "Nome do termo relacionado"
                }
              }
            }
          }
        }
      },
```

```
      "relations": {
        "type": "array",
        "description": "Relações no conteúdo dos documentos da plataforma",
        "items": {
          "type": "object",
          "description": "Relação encontrada",
          "required": ["sentence", "objects"],
          "properties": {
            "sentence": {
              "type": "string",
              "description": "A frase onde a relação se encontra"
            },
            "object": {
              "type": "string",
              "description": "Objeto da frase, considerando que o termo do
                              campo word é o sujeito"
            },
            "relation": {
              "type": "string",
              "description": "A relação entre os dois termos"
            }
          }
        }
      }
    }
  }
}
```

Apêndice D – Information Extraction Applications for Clinical Trials: A Survey

Information Extraction Applications for Clinical Trials: A Survey

Sofia Alves
Polytechnic of Coimbra - ISEC
Coimbra, Portugal
a21230175@isec.pt

Jorge Bernardino
Polytechnic of Coimbra - ISEC
Coimbra, Portugal
jorge@isec.pt

João Costa
Polytechnic of Coimbra - ISEC
Coimbra, Portugal
jcosta@isec.pt

Abstract — **Many Information Extraction (IE) applications are available, in several domains, to handle the increasingly volume of data. This paper reviews applications and methods used for IE and performs an analysis with the objective of understanding which applications have already been used for clinical trials and applications that although they were not and would have an easy adaptation to be used as well.**

Keywords - Information Extraction; Medical Documentations; Clinical Trials; Tools.

I. INTRODUCTION

The information available has grown tremendously in recent years, in a wide range of areas, from social to technical data. Information extraction (IE) applications have been proposed, in several domains, to effectively handle such increasingly volume of data. The goal of this paper is to survey IE applications, in particular those that were used, or can be used, to handle data from clinical trials, produced by hundreds of authors.

Clinical trials are studies that evaluate the effectiveness of new experimental drugs, medications, medical devices or procedures made to groups of people and usually encompass several periods, also called as phases. The phase 0 is usually done to a small number of people (usually less than 15). In this phase, investigators use small dosages of a experimental drug to make sure that is not harmful before it can be used higher dosages in the next phases. On phase I, investigators spend months evaluating the effects of the drug to a larger group of people (between 20 to 80). The objective of this phase is to determine the highest dosage that a human can take before provoking serious side effects. In addition to evaluating the ideal dosage, investigators also look at the best way to administer the drug (orally, intravenously, topically). The following phase - phase II - involves several hundreds of participants that have the condition that the experimental drug is meant to treat. Investigators monitor these participants for several months, or years, to assess the effectiveness of the drug. On phase III, a clinical trial may involve up to 3000 participants who have the condition. The objective of this phase is to evaluate how the new drug compares to with existing medications that treat the condition. Finally, a clinical trial reach phase IV, when the FDA (Food and Drug Administration) approves it. This phase can last for many years and the investigators continue to collect more information about the medication's long-term safety, effectiveness, etc [1].

The produced documents in the clinical trials have unstructured information. In this context, the information extraction assumes an important role. Information Extraction is the field that focuses on extracting useful information using different methods and approaches, such as of Natural Language Processing, and allows that information to be structured and then be treated as intended.

This paper is organized as follows: Section 2 briefly explains the used method for the construction of this paper and covers the basic definitions and relevant concepts of Text Mining and Information Extraction. Section 3 presents a summary of the relevant IE applications used in recent years. Finally, section 4 concludes this study with a discussion about which tools can be applied to extract information from clinical trials.

II. BACKGROUND

Information Extraction IE is one of the Text Mining areas. Text mining refers to the process of extracting information from texts and, based on the characteristics, can be divided into seven practice areas [2].

Search and Information Retrieval (IR) focus on storage and retrieval of text documents from databases. Document clustering and Document Classification are the areas responsible for grouping snippets, paragraphs or documents, using data mining classification methods Document Classification frequently use models trained on labeled examples. Web mining involves data and text mining from the Internet, with focus on the scale and interconnectedness of the web. Natural Language Processing (NLP) focus on low-level languages for processing and understanding tasks. Concept extraction allows the creation of groups of words and phrases into semantically similar groups. Lastly, Information Extraction, that allows the identification and extraction of relevant facts and relationships from unstructured texts [2], such as stories, news, etc. Fig. 1 shows the connection between them.

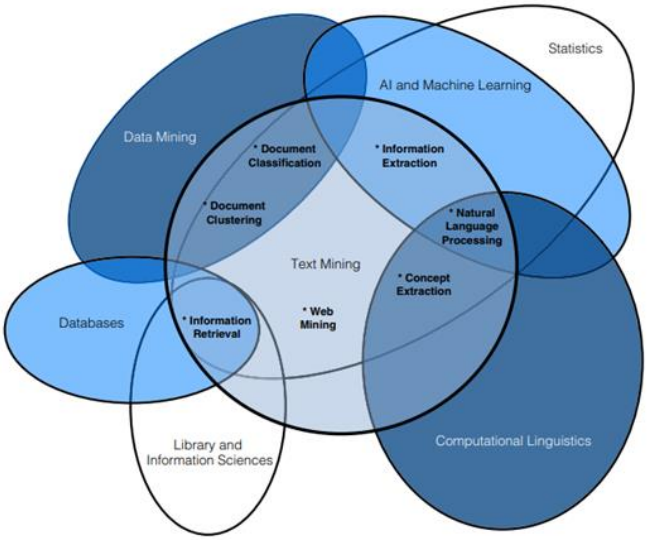

Figure. 1. Connection between Text Mining Areas. Source: [2]

IE applications typically involved a set of sequential tasks. First, the text has to be pre-processed, where is prepared for processing with the help of computational linguistics tools such as tokenization, sentence splitting, morphological analysis, etc. Concepts are extracted (found) and classified concepts – for example, mentions of people, things, locations, events, and others. The related concepts are linked when a relationship between extracted concepts are identified. Afterwards, the extracted data is presented into a standard form and the duplicate data is removed. Finally, the extracted knowledge is usually ingested into a database for later usage. Information Extraction can be entirely automated or performed with the help of human input [3]. For a better perception, see the fig. 2.

Within the IE, there are different approaches and the more relevant are defined below:

1. **Unsupervised learning systems** approach is used for corpus bootstrapping methods that depend on small seed rules that are learned from annotated systems [4]**;**
2. **Supervised learning systems** approach requires many large amounts of data set as training data. The machine learning rules are then used to extract required information [4];
3. **Rule-based systems** are a category of Open IE. Open IE, unlike traditional methods of IE, is not limited to a small set of target relations known in advance. Extracts all types of relations found in a text. So, these systems make use of hand-crafted extraction rules [5].
4. **Clause-Based Systems** is also a category of Open IE. Aimis to improve accuracy and is based on the idea of restructuring the sentence. The goal is to transform complex sentences, with relationships distributed across clauses or presented in a non-canonical way, in a set of syntactically simplified independent clauses that are easy to interpret [5]**.**
5. **Convolutional Neural Network (CNN)** has been applied to Natural Language Processing tasks. In this way, CNN builds features through automatic learning, fitting different domains well instead of building hand-craft features [6].
6. **Ontology-Based Information Extraction (OBIE)** is one of the subfields of Information Extraction. A Ontology is a formal and explicit specification of a conceptualization, which has a crucial role in the process of Information Extraction [7].

The next section, will present some of the developed applications based on mentioned IE approaches.

III. IE APPLICATIONS

This section presents existing methods and IE applications, then figure out which could be used for clinical trials.

In the Open IE systems, [5] provided an overview with some approaches presented above.

Within Learning-based Systems, [5], [8] presented OLLIE (Open Language Learning Information Extraction) was the first Open IE identifying not only verb-based relations but also nouns and adjectives relationships. Making possible to cover a wider range of relationship expressions. TextRunner [9] and WOE [10] are also applications developed with this approach. ReVerb [11] and the most recently: KrakeN [5], [12], Exemplar [13] and PredPatt [14] are examples of rule-based systems.

ClauseIE [5], [15] is a In the Clause-based system that explores the linguist knowledge present in English grammar. That allow mapping dependency relations in a sentence to clause constituents.

Still in the context of Open IE applications, Niklaus et all [5] analyzed NestIE [17], MinIE [18] and Graphene [19].

[16] presents two extensions to the OBIE (Ontology-based information extraction) architecture. The general idea is to use ontologies to guide the information extraction process. These two extensions lead to a hybrid approach for the extraction process. The first extension increases the accuracy of the OBIE process. Does it through the combination of information extractors with different implementation strategies. The second extension adds functionality to identify incorrect elements in the text.

[19] is an implementation of a clinical information extraction tool with the CNN method. This implementation leverages the deep neural network to annotate event spans and their attributes from raw clinical notes and pathology reports.

Wang et al [20] explored tools for extraction of information in clinical information.

According to this article, the most frequently used tools for IE in the clinical domains are cTakes, MetaMap and MedLEE, ordered by frequency of use. Other identified tools are KMCI (KnowlegdeMap Concept Indexer), HITEx, MedEx, Medtex, MedXN, MedTime and the frameworks UIMA, GATE, Protégé and MedTagger.

Firstly, cTakes [20], developed by Mayo Clinic is the most used tool. It is built upon multiple Apache open-source projects, the Apache Unstructured Information Management Architecture (UIMA) framework [20], [21] and Apache OpenNLP [22] toolkit. It contains analysis engines for various linguistics and clinical tasks, such as sentence detection, tokenization, part-of-speech tagging, concept detection, and normalization. This tool has been adopted for identification of patient phenotype cohorts, smoking status extraction, genome-wide association studies, extraction of adverse drug events, detection of medication discrepancies, temporal relation discovery, risk stratification, and risk factor identification from EHRs.

MetaMap [20], [23], [24] is a tool developed by the National Library of Medicine (NLM). The goal of MetaMap is mapping biomedical text to the Unified Medical Language System (UMLS) Metathesaurus or vice versa.

Firstly was developed to improve biomedical text retrieval of MEDLINE and PubMed citations. Later, was improved to process clinical text [20], [25].

MedLEE [19], [22], is the third most used tool. Is one of the earliest clinical NLP systems and is mostly used for pharmacovigilance and pharmacoepidemiology. This system extracts, structures and encodes clinical information from textual patient reports [26]. MedLEE is currently being used on a daily basis to routinely process radiological reports of patients at CPMC (California Pacific Medical Center).

KnowlegdeMap Concept Indexer [20], [27] is an NLP system that identifies biomedical concepts and maps then to UMLS concepts. The KMCI processes documents in three phases:

1. Sentence Identification and Normalization;
2. Concept Identification and NLP techniques;
3. Concept disambiguation.

This tool was developed to extract concepts represented in medical educational texts [27].

HITExt (Healthy Information Text Extraction) [20], [28] is an open-source natural language processing software developed by a group of researchers at the Brigham and Women's Hospital and Harvard Medical School. HITEx is a data-driven application. It has been successfully used to process medical records from two different institutions.

MedEx[20], [29] has been applied in several studies. It is a rule-based system that extracts medication name, strength, route, and frequency. According to the article [20] the system was evaluated on 50 discharge summaries, and an F-measure of 0.93 was obtained. They analyzed two different corpora: 159 clinical notes from Mayo Clinic.

MedTex is NLP system that extracts meaningful information from narrative text to facilitate clinical staff in the decision-making process [20]. Have been developed in partnership with healthcare practitioners from Cancer Registries, hospital radiology and emergency medicine departments. Medtex provides informed decision support by extracting greater value from their clinical narrative reports [31].

Lastly, in the tools domain, we have MedXN - Medication Extraction and Normalization - a tool to extract comprehensive medication information from clinical narratives and normalize it to RxNorm [20], [32].

The method used for this tool was taken medication descriptions in clinical notes that were decomposed into medication name and attributes, latter, separately extracted using RxNorm dictionary lookup and regular expression. Then, each medication name and its attributes were combined together according to the RxNorm convention to find the most appropriate RxNorm representation. For this, it has employed serialized hierarchical steps implemented in Apache Unstructured Information Management Architecture.

In the context of frameworks, the first to be analyzed is UIMA - Unstructured Information Management application [20], [21]. So, the UIM applications are a software system that analyzes large volumes of unstructured information at the same time that consulting structured information sources to discover, organize and present knowledge relevant to the application's end-user. Specifically, UIMA is a software architecture to developing UIM applications. The UIMA architecture defines roles, interfaces and communications of large-grained components essential for UIM applications.

Summing up, UIMA enables applications to be decomposed into components, for example:

1. Language identification;
2. Language-specific segmentation;
3. Sentence boundary detection;
4. Entity detection (person/place names, etc).

Each component implements interfaces defined by the framework and provides self-describing metadata via XML.

GATE [20], [33] is a framework and graphical development environment that enables users to develop components and resources in a robust way. This allows not only to develop successful applications for processing tasks, like Information Extraction but also build and annotate evaluations on the applications generated. The framework can be used to develop applications and resources in multiple languages with Unicode support.

Protégé [20], [34] is a framework that has been frequently used for ontology building. Developers that want to use this framework can integrate the output of Protégé with rule systems or other problem solvers to construct a wide range of intelligent systems. It also includes deductive classifiers to validate that models are consistent and to infer new information based on the analysis of an ontology.

MedTime is a framework to extract temporal information from clinical narratives and normalize it to the TIMEX3 standard [28], [44]. It is hybrid with interweaving rule-based and machine learning procedures for temporal information extraction. It is divided into six steps: pre-processing, temporal tagging by HeidelTime, clinical frequency tagging, sequence labeling, clinical temporal normalization and post-processing.

MedTagger [19], [30], another IE framework that includes three major components: MedTagger for indexing based on dictionaries, MedTaggerIE for information extraction based on patterns, and MedTaggerML for machine learning-based named entity recognition. It is an open-source NLP pipeline based on UIMA framework for indexing based on dictionaries, information extraction, and machine learning–based named entity recognition from clinical text.

TABLE 1
CLINICAL IE TOOLS

Tool	Description	Method	Dataset	Tasks
cTakes [42]	Open-source system based on UIMA framework for extraction of information	NLP	Eletronic Health Records - EHRs	Identification of patient phenotype cohorts; Smoking status extraction; Genome-wide association studies; Extraction of adverse drug events; Detection of medication discrepancies; Temporal relation discovery; Risk stratification; Risk factor identification
MetaMap [43]	Tool that maps biomedical text to UMLS concepts	Natural-language processing (NLP) and computational-linguistic techniques	Process clinical text	Phenotype extraction; Assessment of emergency department use; Drug-disease treatment relationships; Fragment recognition in clinical documents; Extraction of patient-related attributes
MedLEE [44]	System that extracts, structures and encodes clinical information	NLP	Narrative clinical notes	Pharmacovigilance; Pharmacoepidemiology
KnowledgeMap Concept Indexer (KMCI) [27]	System that extracts biomedical concepts and maps them to UMLS concepts	NLP	Medical educational texts (eg. Medical school curriculum)	UMLS Metathesausrus-based tools to navigate through documents
HITex [45]	Open-source tool built on top of the GATE framework for various tasks.	NLP	Summaries and outpatient notes	Diagnoses extraction; Discharge medications extraction; Smoking status extraction
MedEx [46]	System used to recognize drug names, dose, route and frequency	NLP	Free-text clinical records	System to extract mentions of aspirin use and dosage information and pharmacogenetic studies
MedTex [31]	System that extracts meaningful information from narrative text to facilitate clinical staff in the decision-making process	NLP	Clinical Narrative reports	Informed decision support; Facilitate clinical staff in the decision-making process
MedXN [32]	Tool to extract comprehensive medication information from clinical narratives and normalize it to RxNorm	NLP	Medication descriptions in clinical notes	Medication dictionaries and rules for attribute patterns
UIMA [21]	Software framework for the analysis of unstructured contents	Natural Language Processing (NLP), Information Retrieval (IR), machine learning, and ontologies	Unstructured contents like: text, video and audio data	Search applications
GATE [47]	Java-based open-source software for various NLP tasks	NLP	Unstructured contents from multiple languages	Semantic annotation
Protégé [48]	Open-source ontology editor and framework for building intelligent systems			Ontology building; clinical decision-support systems
MedTagger [30]	Open-source NLP pipeline based on UIMA framework for indexing based on dictionaries, information extraction, and machine learning–based named entity recognition	NLP	Clinical text	Aggregation and labeling huge datasets
MedTime [49]	A tool to extract temporal information from clinical narratives and normalize it to the TIMEX3 standard	NLP	Clinical narratives	Temporal Information Extraction

Some NLP applications, such as NLKT [35] or OpenNLP [22], were used for preprocessing tasks (sentence boundary detection, tokenization, etc). MALLET [36] and WEKA [37] use machine learning algorithms (classification, clustering, etc). TextHunter [38], KneeTex [39], Textractor [40] and NOBLE [41] are other tools found but not included in this review.

The most used clinical IE tools are summarized in Table 1.

IV. Conclusions

This paper presents an overview of the various methods and applications of information extraction.

Some of the applications use a specific extraction method, while others use several methods together to achieve better results, depending on the purpose. Each of them has some specifications and rules that make them more appropriate to a given context.

In general, most of the applications could be adapted to extract information from clinical trials, however, some of these would be more difficult, since the context is different. For example, Graphene is an application that needs to set manually and there is no record of having already been applied to clinical trials, CAINES is a system focused on reports of terrorist accidents, which makes it unfeasible to be adapted to the intended context. Not because it can not be adjusted, but this system has not been tested or prepared for clinical information. The applications mentioned in Table 1, in addition to being the most used, focus on clinical information, which is an advantage. Any of the frameworks mentioned in the Table could be a basis for an information extraction application, which is the case with of cTakes and MedTagger, which are built on top of the UIMA framework. HITex and KnowledgeMAP Concept Indexer are also viable applications to use since the keywords they extract are related to clinical concepts, however, the datasets used are not closer to clinical trials. MetaMap, MedLEE, MedEx, and MedTex are the most viable applications to be used since the dataset used in them is similar to clinical trials.

References

[1] "Clinical Trial Phases: What Happens in Phase 0, I, II, III, and IV," Healthline, 22-Feb-2018. [Online]. Available: https://www.healthline.com/health/clinical-trial-phases. [Accessed: 28-Jan-2019].

[2] "The Seven Practice Areas of Text Analytics," in Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications, Elsevier, 2012, pp. 29–41.

[3] "What is Information Extraction?," Ontotext. .

[4] M. Abdelmagid, A. Ahmed, and M. Himmat, "INFORMATION EXTRACTION METHODS AND EXTRACTION TECHNIQUES IN THE CHEMICAL DOCUMENT'S CONTENTS: SURVEY," vol. 10, no. 3, p. 6, 2015.

[5] C. Niklaus, M. Cetto, A. Freitas, and S. Handschuh, "A Survey on Open Information Extraction," ArXiv180605599 Cs, Jun. 2018.

[6] C. Liu, W. Sun, W. Chao, and W. Che, "Convolution Neural Network for Relation Extraction," in Advanced Data Mining and Applications, 2013, pp. 231–242.

[7] R. Shah and S. Jain, Ontology-based Information Extraction: An Overview and a Study of different Approaches. .

[8] Mausam, M. Schmitz, S. Soderland, R. Bart, and O. Etzioni, "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, Jeju Island, Korea, 2012, pp. 523–534.

[9] M. Banko, M. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open Information Extraction from the Web," IJCAI, vol. 7, pp. 2670–2676.

[10] F. Wu and D. S. Weld, "Open Information Extraction Using Wikipedia," in Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 2010, pp. 118–127.

[11] A. Fader, S. Soderland, and O. Etzioni, "Identifying Relations for Open Information Extraction," in Proceedings of the Conference on Empirical Methods in Natural Language Processing, Stroudsburg, PA, USA, 2011, pp. 1535–1545.

[12] A. Akbik and A. Löser, "KrakeN: N-ary Facts in Open Information Extraction," in Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX), Montrèal, Canada, 2012, pp. 52–56.

[13] J. Schmidek and D. Barbosa, "Improving Open Relation Extraction via Sentence Re-Structuring," p. 4.

[14] N. Bhutani, H. V. Jagadish, and D. Radev, "Nested Propositions in Open Information Extraction," in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, 2016, pp. 55–64.

[15] M. Cetto, C. Niklaus, A. Freitas, and S. Handschuh, "Graphene: Semantically-Linked Propositions in Open Information Extraction," ArXiv180711276 Cs, Jul. 2018.

[16] F. Gutierrez, D. Dou, S. Fickas, D. Wimalasuriya, and H. Zong, "A hybrid ontology-based information extraction system," J. Inf. Sci., vol. 42, no. 6, pp. 798–820, Dec. 2016.

[17] P.-Y. Chen, Y.-H. Lee, Y.-H. Wu, and W.-Y. Ma, "IExM: Information Extraction System for Movies," 2017, pp. 189–193.

[18] P. N. Golshan, M. Student, and L. Safari, "A Study of Recent Contributions on Information Extraction," p. 6.

[19] P. Li and H. Huang, "Clinical Information Extraction via Convolutional Neural Network," ArXiv160309381 Cs, Mar. 2016.

[20] Y. Wang et al., "Clinical information extraction applications: A literature review," J. Biomed. Inform., vol. 77, pp. 34–49, Jan. 2018.

[21] D. Ferrucci and A. Lally, "UIMA: an architectural approach to unstructured information processing in the corporate research environment," Nat. Lang. Eng., vol. 10, no. 3–4, pp. 327–348, Sep. 2004.

[22] "Apache OpenNLP," 14-Dec-2018. [Online]. Available: https://opennlp.apache.org/. [Accessed: 14-Dec-2018].

[23] A. R. Aronson and F.-M. Lang, "An overview of MetaMap: Historical perspective and recent advances," J. Am. Med. Inform. Assoc., vol. 17, no. 3, pp. 229–236, 2010.

[24] "MetaMap - A Tool For Recognizing UMLS Concepts in Text," 14-Dec-2018. [Online]. Available: https://metamap.nlm.nih.gov/. [Accessed: 14-Dec-2018].

[25] A. R. Aronson, J. G. Mork, A. Névéol, S. E. Shooshan, and D. Demner-Fushman, "Methodology for Creating UMLS Content Views Appropriate for Biomedical Natural Language Processing," AMIA. Annu. Symp. Proc., vol. 2008, pp. 21–25, 2008.

[26] C. Friedman, G. Hripcsak, W. DuMouchel, S. B. Johnson, and P. D. Clayton, "Natural language processing in an operational clinical information system," Nat. Lang. Eng., vol. 1, no. 1, pp. 83–108, Mar. 1995.

[27] J. C. Denny, P. R. Irani, F. H. Wehbe, J. D. Smithers, and A. Spickard, "The KnowledgeMap Project: Development of a Concept-Based Medical School Curriculum Database," AMIA. Annu. Symp. Proc., vol. 2003, pp. 195–199, 2003.

[28] S. Goryachev, M. Sordo, and Q. T. Zeng, "A Suite of Natural Language Processing Tools Developed for the I2B2 Project," AMIA. Annu. Symp. Proc., vol. 2006, p. 931, 2006.

[29] J. Cowie and W. Lehnert, "Information Extraction," Commun ACM, vol. 39, no. 1, pp. 80–91, Jan. 1996.

[30] C. Zheng, N. Rashid, R. Koblick, and J. An, "Medication Extraction from Electronic Clinical Notes in an Integrated Health System: A Study on

Aspirin Use in Patients with Nonvalvular Atrial Fibrillation," Clin. Ther., vol. 37, no. 9, pp. 2048-2058.e2, Sep. 2015.

[31] "MedTex - Medical Text Processing Software," AEHRC, 23-Nov-2017.

[32] S. Sohn, C. Clark, S. R. Halgrim, S. P. Murphy, C. G. Chute, and H. Liu, "MedXN: an open source medication extraction and normalization tool for clinical text," J. Am. Med. Inform. Assoc. JAMIA, vol. 21, no. 5, pp. 858–865, Oct. 2014.

[33] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, "GATE: an Architecture for Development of Robust HLT applications," in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, 2002.

[34] M. A. Musen, "The ProtÉGÉ Project: A Look Back and a Look Forward," AI Matters, vol. 1, no. 4, pp. 4–12, Jun. 2015.

[35] S. Bird, "NLTK: The Natural Language Toolkit," presented at the In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics, 2002.

[36] "MALLET: A machine learning for language toolkit," ResearchGate. [Online]. Available: https://www.researchgate.net/publication/220017648_MALLET_A_machine_learning_for_language_toolkit. [Accessed: 01-Feb-2019].

[37] G. Holmes, A. Donkin, and I. H. Witten, "WEKA: a machine learning workbench," in Proceedings of ANZIIS '94 - Australian New Zealnd Intelligent Information Systems Conference, Brisbane, Qld., Australia, 1994, pp. 357–361.

[38] R. G. Jackson MSc, M. Ball, R. Patel, R. D. Hayes, R. J. B. Dobson, and R. Stewart, "TextHunter – A User Friendly Tool for Extracting Generic Concepts from Free Text in Clinical Research," AMIA. Annu. Symp. Proc., vol. 2014, pp. 729–738, Nov. 2014.

[39] I. Spasić, B. Zhao, C. B. Jones, and K. Button, "KneeTex: an ontology–driven system for information extraction from MRI reports," J. Biomed. Semant., vol. 6, no. 1, p. 34, Sep. 2015.

[40] S. Meystre, J. Thibault, S. Shen, J. F Hurdle, and B. South, "Textractor: A hybrid system for medications and reason for their prescription extraction from clinical text documents," J. Am. Med. Inform. Assoc. JAMIA, vol. 17, pp. 559–62, Sep. 2010.

[41] "NOBLE - Flexible concept recognition for large-scale biomedical natural language processing. - PubMed - NCBI." [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/26763894. [Accessed: 01-Feb-2019].

[42] "Apache cTAKESTM - clinical Text Analysis Knowledge Extraction System." [Online]. Available: http://ctakes.apache.org/. [Accessed: 06-Feb-2019].

[43] "MetaMap - A Tool For Recognizing UMLS Concepts in Text." [Online]. Available: https://metamap.nlm.nih.gov/. [Accessed: 06-Feb-2019].

[44] C. Friedman, L. Shagina, S. A. Socratous, and X. Zeng, "A WEB-Based Version of MedLEE: A Medical Language Extraction and Encoding System," Proc. AMIA Annu. Fall Symp., p. 938, 1996.

[45] "HITEx Manual." [Online]. Available: https://www.i2b2.org/software/projects/hitex/hitex_manual.html. [Accessed: 06-Feb-2019].

[46] "MedEx - Resources - Center for Computational Biomedicine - The University of Texas Health Science Center at Houston (UTHealth) School of Biomedical Informatics." [Online]. Available: https://sbmi.uth.edu/ccb/resources/medex.htm. [Accessed: 06-Feb-2019].

[47] "GATE.ac.uk - index.html." [Online]. Available: https://gate.ac.uk/. [Accessed: 06-Feb-2019].

[48] "protégé." [Online]. Available: https://protege.stanford.edu/. [Accessed: 06-Feb-2019].

[49] Y.-K. Lin, H. Chen, and R. A. Brown, "MedTime: a temporal information extraction system for clinical narratives," J. Biomed. Inform., vol. 46 Suppl, pp. S20-28, Dec. 2013.