UNIVERSIDADE DA BEIRA INTERIOR
Engenharia

Métodos de Deteção Automática de Plágio Extrínseco em Textos de Grande Dimensão

Joaquim Domingos Mussandi

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor. João Paulo da Costa Cordeiro

Covilhã, Novembro de 2018

Dedicatória

Ao meu pai Lucas João Jorge (em memória), que não tive o privilégio de honrar o seu ato fúnebre com a minha presença.
Ao meu filho Jessé Mussandi que não o dei carinho devido nos primeiros meses.
Ao Lucas Júnior e as divas da minha vida Graça Mussandi, Alice Manuel e Albertina Domingos por tudo que vocês representam para mim, amo-vos.

Agradecimentos

Em primeiro lugar quero agradecer à Deus por tudo, porque sem Ele nada sou.
De seguida, expresso o meu profundo apreço, aos meus familiares, especialmente, os meus irmãos pelo carinho e atenção que me têm prestado de forma incomensurável.
Aos professores da UBI do curso de Engenharia Informática que lecionaram as unidades curriculares que tive neste curso de mestrado, com particular realce ao meu orientador, Professor Doutor João Paulo Cordeiro pela atenção, suporte e reconhecida dedicação na orientação desta dissertação.
À minha gratidão à reitoria da Universidade Kimpa Vita (UNIKIV), à decana da ESPU/UNIKIV, Professora Doutora Maria de Fátima e aos seus vices-decanos;
Aos meus colegas, particularmente aqueles com quais formei grupos nas várias unidades curriculares, designadamente o Moisés Ferreira, Ricardo Oliveira, Jusualdo Figueira e João Silva. Este último pela disponibilidade e suporte oferecido.
Aos meus companheiros do laboratório *HULTIG* Fátima Dantas, Domingos Dionísio, Mvita Zankulu e Ramos Pedro.
Aos meus amigos António Kagibebi, Ferraz Quicongo e Mizele Coxe; os membros da Public Campus Ministries - UBI, a Igreja Adventista do Sétimo dia do Fundão, a ADRA-Fundão, ao INAGBE.
Aos meus companheiros de viagem Dalne António, Ngombo Armando e Luzizila Salambiaku e todos aqueles que, de uma forma ou de outra, contribuíram para que este sonho se tornasse realidade.

Agradeço!

Resumo

A prática de plágio em documentos, livros e na arte de forma geral, tem consequência gravas na sociedade. A existência de pessoas sem honestidade, na academia, na indústria, na imprensa que se apropriam da propriedade intelectual de outrem, levou algumas organizações a produzirem normas de combate ao plágio e adotarem meios tecnológicas para enfrentar e evitar a propagação deste mal.
Os sistemas de Deteção Automática de Plágio (DAP) são, sem dúvida, os principais meios utilizadas para identificação de situações que envolvem a prática de plágio em documentos de texto disponíveis na *Web*.
Para tentar ofuscar a atitude fraudulenta (omitir o plágio) em um documento de texto de grande dimensão, os praticantes de plágio, algumas vezes extraem curtas frases, sendo consequentemente manipuladas e transformadas de voz ativa para passiva e vice-versa, bem como os léxicos transformados em sinónimos e antónimos [ASA12, AIAA15, ASI+17]. Por outra, com pares de texto[1] de maior tamanho, o processo de *alinhamento textual* é fastidioso, que o torna menos eficiente e até menos eficaz, sobretudo, se existir tentativa de ofuscação.
Este trabalho tinha como objetivo propor métodos de DAP menos complexos que tornam o processo da *Análise Detalhada* mais *eficiente* e com melhor *eficácia*. Para tal, desenvolvemos dois métodos de DAP primeiramente, um método de deteção plágio que utiliza uma abordagem de *segmentação recursiva do documento fonte em três blocos*, afim de identificar pequenos e grandes segmentos plagiados com paráfrases com eficácia e alto nível de eficiência temporal.
O segundo método proposto é o de *Pesquisa de Plágio por Scanning Vetorial*. Este método utiliza *word embeeding* (*word2vec*) sem recurso aos cálculos matriciais, e é capaz de detetar quer pequenos segmentos plagiados, quer segmentos grandes, mesmo com alto nível de ofuscação de forma eficiente e com alto nível de eficácia.
Os resultados que apresentados no Capítulo 4 demonstram a eficácia e a eficiência dos métodos propostos nesta dissertação.

Palavras-Chave

Deteção Automática de Plágio Extrínseco, Recuperação de Informação, Similaridade documental, Análise-Detalhada, plágio-word2vec.

[1] pares de texto: segmentos extraídos do documento fonte em comparação com outro segmento extraído do documento suspeito.

Abstract

The practice of plagiarism in documents, books, and art in general has repercussions on society. The existence of people without honesty, in the academy, in the industry, in the press that appropriates the intellectual property of others, led some organizations to produce norms to combat plagiarism and to adopt technological means to confront and to prevent the propagation of this evil. Plagiarism Automatic Detectiors (PAD) systems are undoubtedly the main means used to identify situations involving the practice of plagiarism in text documents available in *Web*.
To attempt to obfuscate the fraudulent attitude (omitting plagiarism) in a large text document, plagiarists sometimes extract short phrases and are consequently manipulated and transformed from active to passive and vice versa, as well as lexicons transformed into synonyms and antonyms [ASA12, AIAA15, ASI+17]. On the other, with pairs of text [2] Of larger size, the process of *text alignment* is tedious, which makes it less efficient and even less effective, especially if there is an attempt to obfuscate.
This work aimed to propose less complex PAD methods that make the *Detailed Analysis* process more *efficient* and with better *efficiency*. For this, we developed two methods of PAD first, a plagiarism detection method that uses a recursive segmentation approach of the source document in three blocks, in order to identify small and large segments plagiarized with efficacious paraphrases and high level of temporal efficiency. The second proposed method is the *Plagiarism Research by Vector Scanning)*. This method uses *word embeedings* (*word2vec*) without recourse to matrix calculations, and is capable of detecting either small plagiarized segments or large segments, even with high level of obfuscation efficiently and with high level of efficiency. The results presented in Chapter 4 demonstrate the efficacy and efficiency of the methods proposed in this dissertation.

Keywords

Automatic Detection of Extrinsic Plagiarism, Information Retrieval, Documentary Similarity, Analysis-Detailed, plagiarism-word2vec.

[2] text pairs: segments extracted from the source document compared to another segment extracted from the suspicious document.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

APEE Associação Portuguesa de Estudos Europeus

API Application Programming Interface

BOW Bag-Of-Words

CBOW Continuous Bag-Of-Word

CLEF Conference and Labs of the Evaluation Forum

CPU Central Processing Unit

DAP Deteção Automática de Plágio

EUCD European Union Copyright Directive

IDE Integrated Development Environment

IEEE Institute of Electrical and Electronics Engineers

JRB Métrica de Jaccard Recursiva Binária

JRT Métrica de Jaccard Recursiva Ternária

LDA Latent Dirichlet Allocation

LSA Latent Semantic Analysis

LSI Latent Semantic Indexing

MRB Métrica Recursiva Binária

MJRB Métrica de Jaccard Recursiva Binária

MRT Métrica de Jaccard Recursiva Ternária

nBOW Normalized Bag-Of-Words

OMPI Organização Mundial de Propriedade Inteletual

PAD Plagiarism Automatic Detectiors

PAN Plagiarism Analysis Authorship Identification and Near-Duplicate Detection

PLN Processamento da Linguagem Natural

PSVS Plagiarism Search by Vector Scanning

POS Part-Of-Speech

PSD Positive SemiDefinite

PPSV Pesquisa de Plágio por Scanning Vetorial

RI Recuperação de Informação

RWMD Relaxed Word Moving Distance

SIF Finding all Similar Files in a large file system

SPA Sociedade Portuguesa de Autores

TF-IDF Term Frequency–Inverse Document Frequency

URL Uniform Resource Locator

VSM Vector Space Model

WMD Word Mover's Distance

Capítulo 1

Introdução

1.1 Motivação e Foco

O plágio é um mal que afeta todas as esferas da vida social. A falta de honestidade por parte de certas pessoas ligadas à indústria, academia, imprensa que, se apropriam da propriedade intelectual de outrem, levou as universidades, organizações nacionais como Sociedade Portuguesa de Autores (SPA), Associação Portuguesa de Estudos Europeus (APEE) e organismos internacionais tais como a Organização Mundial de Propriedade Inteletual (OMPI) e a União Europeia a adotarem ferramentas como pauta deontológica, Diretiva da União Europeia sobre Direito de Autor (do inglês - European Union Copyright Directive (EUCD)), para punição aos infratores destas normas. Mas, pensar nas normas como os únicos meios para o combate ao plágio, não é de todo viável. Por isso, juntam-se esforços e faz-se recurso aos meios tecnológicos. Hoje, até com um *smartphone*, pode se recorrer as ferramentas de Deteção Automática de Plágio (DAP) [NB17]. Estas ferramentas são os principais recursos utilizados para identificação de casos de plágio.

As manchetes nos órgãos de informação nos brindam com letras garrafais com notícias envolvendo políticos a terem os seus títulos académicos retirados [uol12, Gir13, Onl15], autores que defraudam [LR11, Onl15], universidades prestigiadas com inquéritos judiciais abertos [Kro11] e outras em final de inquérito para submeter ao ministério público [tvi17]. São várias manifestações de ilegitimidade à autoria.

Por vezes, num documento de texto de grande dimensão extrai-se uma curta frase e/ou várias pequenas frases, sendo consequentemente manipuladas e transformadas de voz ativa para passiva e vice-versa, bem como os léxicos transformados em sinónimos e antónimos para ofuscar a atitude fraudulenta [ASA12, AIAA15, ASI+17]. Por outra, com pares de texto de maior tamanho, o processo de alinhamento textual é fastidioso, que o torna menos eficiente e até menos eficaz, sobretudo, se existir tentativa de ofuscação.

Este trabalho tinha como objetivo propor métodos de DAP menos complexos que tornam o processo da *análise detalhada* mais *eficiente* e com melhor *eficácia*. Para tal, desenvolvemos dois métodos de DAP primeiramente, um método de deteção plágio que utiliza uma abordagem de *segmentação recursiva do documento fonte em três blocos*, a fim de identificar pequenos e grandes segmentos plagiados com paráfrases, de forma eficaz e com alto nível de eficiência temporal. O segundo método proposto é o de *Pesquisa de Plágio por Scanning Vetorial* (PPSV). Este método utiliza *word embbedings* (*word2vec*) sem recurso aos cálculos matriciais, e é capaz de detetar pequenos segmentos plagiados ou segmentos enormes, mesmo com alto nível de ofuscação de maneira eficiente e com alto nível de eficácia.

Os resultados que apresentamos no Capítulo 4 demonstram a eficácia e a eficiência dos métodos propostos nesta dissertação.

1.2 Definição do Problema e Objetivo

Em muitas esferas da vida social existem várias manifestações de ilegitimidade à autoria. Por exemplo, na imprensa, as mesmas ideias são expressas utilizando palavras diferentes e em alguns casos sem dar devido crédito ao seu autor. Isso ocorre também ao nível da academia quando alunos apropriam-se de ideias de autores conhecidos ou de outros alunos de outras universidades que tenham feito trabalhos similares, e fazem-nas passar por suas.
Para descobrir se um documento ou parte dele foi plagiado de uma outra qualquer fonte textual - documento(s) fonte, a tarefa é desafiadora. Torna-se ainda mais desafiadora quando em um documento enorme se extraí um segmento de grande dimensão e manipula-se o mesmo para omitir o plágio. Nesse caso, a procura perde eficiência no processo de DAP.
Neste trabalho, o objetivo era propor dois métodos de *análise detalhada de documentos*, menos complexos e capaz de identificar características textuais e, tornar este processo mais eficiente quer com pequenos segmentos de texto, quer com segmentos enormes com plágio extraídos em documentos de grande dimensão, parafraseado e/ou com alto nível de ofuscação, identificando-os de maneira eficaz. Isso nos levou a criar um sistema de DAP extrínseco que funciona baseando-se nos métodos propostos.

1.3 Principais Contribuições

Embora muito já se fez ao nível da DAP, à medida que o tempo passa, novos *modus operandi* são descobertos e novas pesquisas devem ser feitas para combater estas práticas. De um modo geral, as principais contribuições da nossa pesquisa são:

- nova técnica de criação de vocabulários;
- utilização da recursividade como forma de análise detalhada de documentos para deteção paráfrases eficientemente e com maior eficácia;
- criação de um Algoritmo de DAP otimizado, utilizando *word embbedings* (*word2vec*) sem recursos aos cálculos matriciais.

De forma genérica, nosso principal contributo é aumentar eficácia em cada um dos métodos propostos com alto nível de eficiência em situações de plágio com pequenos ou grandes segmentos manipulados.

1.4 Estrutura da Dissertação

A dissertação está estruturada da seguinte forma:

- Capítulo 1: abordamos os aspetos introdutórios da nossa pesquisa. Falámos do foco da pesquisa, do problema e dos objetivos a alcançar;
- Capítulo 2: neste capítulo, fazemos uma incursão sobre a literatura e apresentámos o enquadramento e o estado da arte sobre a DAP.
- Capítulo 3: apresentámos os métodos de análise detalhada e as métricas implementados nesta dissertação.

- Capítulo 4: apresentámos os resultados obtidos, fazemos uma discussão sobre os mesmos numa análise comparativa com os resultados existentes na literatura.
- Capítulo 5: apresentámos as conclusões alcançadas com a nossa pesquisa, salientando alguns aspetos mais relevantes e perspetivámos desafios para os trabalhos futuros.

Capítulo 2

Enquadramento e Estado da Arte

Neste capítulo, fazemos uma análise sucinta sobre o estado da arte da DAP, observando as principais conferências da área, com especial atenção a *Plagiarism Analysis Authorship Identification and Near-Duplicate Detection (PAN)*, (em português, Análise de Plágio, Identificação da fonte, e Deteção de Duplicados Aproximados), que desde 2009 realiza suas edições anuais. Apresentamos as particularidades dos assuntos tratados. Fazemos alguns enquadramentos conceptuais em síntese, para situar o leitor. Em outros momentos, especificamos detalhes relativamente ao foco da nossa pesquisa.
Na Secção 2.1 começámos com aspetos genéricos sobre o plágio, posteriormente um enquadramento histórico de forma cronológica das tendências das pesquisas na área, alguns métodos de *recuperação de documentos fonte* e alguns métodos de *análise detalhada de documentos* para DAP.

2.1 Generalidade

A prática de plágio em documentos digitais já existe há algumas décadas e a preocupação em combater este delito é relativamente recente [SKS07, PSBCR10, PEBc$^+$11, ASA12]. Não obstante, vários pesquisadores têm investido diversos recursos para combater esta prática criando e/ou reutilizando técnicas, métodos, heurísticas, algoritmos e desenvolvendo ferramentas de DAP. Foi com esta preocupação que em 1993, Manber [Man93] desenvolveu uma ferramenta para deteção de similaridade entre ficheiros que ficou conhecida como *Finding all Similar Files in a large file system (SIF)*. O objetivo do SIF era fazer comparações para identificar se os ficheiros tinham a mesma fonte ou tinham partes provenientes da mesma fonte, além da gestão de ficheiros e a deteção de cópias não autorizadas.
Muhr *et al.* [MZKG09, LHS$^+$12, HEB13, ASP15, AIAA15, AS16, FSGRB16, NSC17] descrevem o plágio como sendo a cópia deliberada de uma informação existente, tal como em documento de texto (quer seja na mesma língua ou em outra (*Cross-language Plagiarism Detection*)), ou *software*. Isto pode incluir a reutilização de material por terceiros sem autorização e/ou citação do autor. Já Potthast *et al.* [PBcE$^+$10, PEBc$^+$11, PHG$^+$12, PHG$^+$13, PHB$^+$14, FC16], afirmam que, estamos diante de um caso de plágio quando $s = (s_{plg}, d_{plg}, s_{src}, d_{src})$, onde s_{plg} representa o segmento de d_{plg}, e d_{plg} é o documento com plágio, s_{src} o segmento do documento fonte oriundo de d_{src} que é o documento plagiado, normalmente indexado numa coleção de documentos (*corpus*). Todos os documentos fonte candidatos a plágio são recuperados a partir da coleção. Isto é, $d_{src} \subseteq D$.
Genericamente, o plágio é dividido em dois tipos: *plágio intrínseco* e *plágio extrínseco* [SM09, Sta09, HEB13, NLM15, AIAA15, ASI$^+$17]. No **plágio intrínseco** o detetor de plágio atenta para detetar segmentos plagiados baseado exclusivamente na informação proveniente do documento suspeito tal como a análise do estilo de escrita quando o documento suspeito tem mais de dois autores [PBcE$^+$10, ASA12]. Já no **plágio extrínseco** [PSE$^+$09, MZKG09, PBcE$^+$10, ASA12,

AIAA15, NLM15, ASI+17], dado um documento suspeito, e uma coleção de documentos fonte, a ideia é procurar na coleção um ou mais documentos que sejam similares ao documento suspeito ou um ou mais dos seus segmentos encontrem *matches* (correspondência aproximada ou exata) no documento suspeito. Este é o âmbito da nossa pesquisa. Pode-se ver mais em [GGP09, BMB+09, PBcE+10, PSBCR10, ASA12, HEB13, SB14, AIAA15, HPA+17, ASI+17].
Potthast *et al*. [PBcE+10, PEBc+11, PHG+12, PHB+14, HPA+17], apresentam a deteção de plágio extrínseco em três etapas, designadamente: *source retrieval* (recuperação de documentos fonte), *detailed analysis* (análise detalhada ou alinhamento de texto) e *knowledge-based post-processing* (pós-processamento baseado em conhecimento) como ilustra a Figura 2.1.

Na primeira etapa, diminui-se o espaço de pesquisa ficando unicamente com os documentos potencialmente vítimas de plágio, a partir de uma coleção maior;
Na segunda etapa, cada segmento suspeito de plágio é comparado com todos os segmentos potenciais fonte de plágio (foi aqui que o nosso trabalho se concentrou);
Na terceira e última etapa, trata da confirmação dos resultados apresentados pelo *detector de plágios*. Os segmentos dos documentos suspeito e do fonte são apresentados em *pares* para serem visualizados por "humano" a fim de confirmar se se trata de plágio ou é falso positivo (segmento detetado como plágio quando de facto é).
O detetor de plágio, reporta alegado caso de plágio utilizando a expressão $r = (r_{plg}, d_{plg}, r_{src}, d'_{src})$ em que r denota caso de plágio reportado; o r_{plg} é o segmento de plágio contido em d_{plg}, e r_{src} é a fonte de s_{src} em d'_{src} [PBcE+10, PSBCR10]. Em [PBcE+10] afirma-se r detetou s se e somente se $d_{plg} \bigcap r_{plg} \neq 0$, $s_{src} \bigcap r_{src} \neq 0$ e $d_{src} = d'_{src}$. A sub-coleção de documentos candidatos é que designada por s_{src} e d'_{src} é o documento que tem seus segmentos plagiados.

Em [FC15] é nos apresentado a deteção de plágio em duas etapas. Num primeiro momento, através de um motor de pesquisa de código aberto, faz-se a indexação dos documentos fonte na coleção e, posteriormente, com as *keywords* (palavras-chave) e as características textuais, efetua-se a recuperação de documentos candidatos a terem sido plagiados, a partir da coleção (conjunto mais reduzido de documentos, d_{src}). No segundo momento, identificam-se os segmentos iguais entre d_{plg} e d_{src}, contendo tentativas de ofuscação ou reordenação das palavras-chave, ou ainda substituição de palavras pelos seus sinónimos.

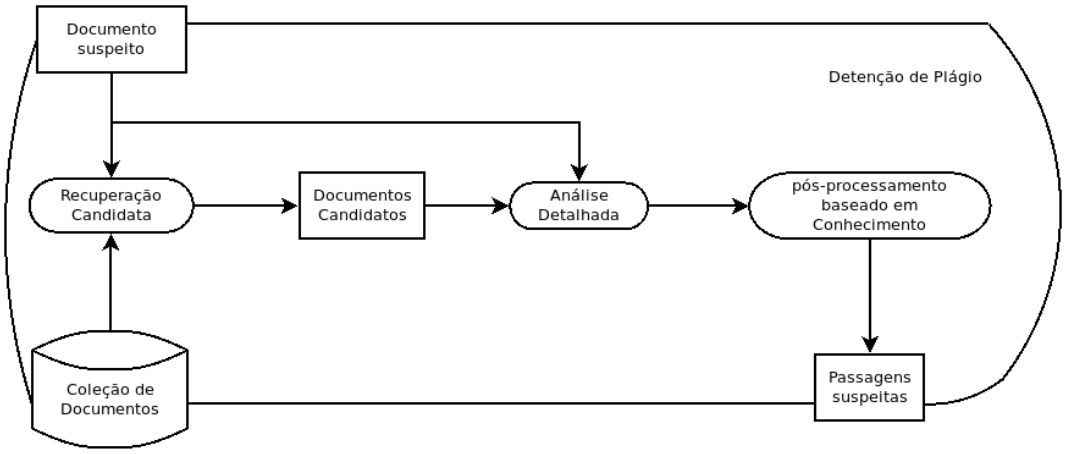

Figura 2.1: Processo de deteção de plágio em três etapas segundo [HPS15].

Abdi *et al*. [AIAA15, ASI+17] apresentam a deteção de plágio numa abordagem composta por

quatro etapas. Na primeira etapa realizam algumas técnicas de Processamento da Linguagem Natural (PLN) nos documentos suspeito e fonte, como é o caso do *pré-processamento* e indexação. Na segunda etapa, reduzem o espaço de procura fazendo a *recuperação de documentos fonte*. Enquanto que na terceira etapa é feita a *análise detalhada* entre os pares de segmentos de ambos documentos (fonte e suspeito) para depois na quarta e última etapa determinar todos os pares de ambos os documentos que são efetivamente segmentos plagiados, designando-se esta última por *pós-processamento*.

De um modo geral, a literatura regista tendências em determinados momentos que, descrevemos-às de seguida:

Stein *et al*. [SKS07], realizaram em 2007, um *workshop* designado por PAN, onde perspetivaram pesquisas em torno de tópicos futuros orientados a:

1. *Plagiarism Analysis* (análise de plágio), deteção de plágio em documentos de texto (plágio intrínseco e extrínseco);
2. *Authorship identification* (identificação do autor) dividido em duas áreas distintas:
   (a) *Authorship Attribuition Problem* (problema de atribuição da autoria), dado um conjunto de autores, o desafio é identificar qual entre esses é o autor de um determinado *texto anónimo* mas, da autoria de um deles.
   (b) *Authorship Verification Problem* (problema de verificação da autoria), dado um autor e um documento de texto o desafio é identificar se este documento é ou não de sua *autoria*.
3. *Near-duplicate detection* (deteção de quase duplicados), este problema tinha haver com baixa precisão na *recuperação de informação* na altura. Tarefa ligada diretamente com a deteção de plágio já que é a primeira etapa do processo de *deteção de plágio extrínseco*.

Em 2008, Barrón-Cedeño e Rosso [CR08] desenvolveram uma abordagem de deteção de plágio com referência, utilizando um *modelo estatístico da linguagem* como forma de tentar explorar os níveis gramaticais e a *perplexidade* para detetar características textuais relevantes nos segmentos plagiados.

Em 2009, [MZKG09, GGP09, BMB+09, Sta09, PSE+09] tipificaram duas áreas de deteção de plágio: *o plágio extrínseco* e *plágio intrínseco*. No plágio extrínseco, dado um documento suspeito e uma coleção fonte, a tarefa passa por identificar os segmentos do documento suspeito que são plágios, e as suas secções de origem, no documento fonte [PSE+09].
Cedeño *et al*. [BCRB09] motivados pelo facto de que, algumas vezes, o segmento que contém plágio é modificado e misturada entre os outros, ou ainda, pode ser uma combinação de vários frases do documento fonte. Comparar documentos inteiros ou suas frases podem não resultar como inicialmente esperado. Por esta razão, apresentam uma abordagem flexível. Os autores codificaram as frases suspeitas e o documento fonte em *n-gram*, sem que o documento fonte fosse dividido. Em particular, utilizaram o *tri-gram* para identificar se a i-ésima frase suspeita era proveniente do documento fonte. Esta abordagem, designada por *kullback-leibler symmetric distance*, apresentou uma redução drástica do espaço de procura.

No 2010 e 2011, [PBcE+10, PSBCR10, PEBc+11] definiram-se os *standards* do que é *caso de plágio*, *relato de um caso de plágio* e reforçou-se a utilização das medidas *plagdet, precision,*

recall, *accuracy*, e *granularity* (abordadas na Secção 4.1), para aferir a performance das ferramentas de deteção de plágio. Em [PBcE+10], tipicamente, a deteção de plágio é nos apresentada em três etapas: *recuperação de documentos fonte*, *análise detalhada de documento* e *pós-processamento baseado no conhecimento* (referidos nas secções 2.2 e 2.4 respetivamente).

Em 2012, Potthast *et al.* [PHG+12], apresentam uma abordagem de deteção de plágio extrínseco em duas fases distintas: (i) *recuperação de documentos fonte* e (ii) *análise detalhada de documento*. O objetivo dos autores foi aumentar os desafios dos limites da performance dos detetores de plágio, tornando-o mais realístico em relação à competição passada, PAN11.
Neste mesmo ano, Alzahrani *et al.* [ASA12] taxonomiam o plágio em dois tipos principais a saber: **plágio literal** e **plágio inteligente**. No primeiro temos cópia exata, cópia aproximada e cópia modificada, enquanto que no segundo temos manipulação de textos (que inclui o parafraseado e a sumarização), tradução de textos (automático ou manual) e fazer-se passar por autor ou proprietário da ideia apresentada.
Na competição da PAN13 [PHG+13], a DAP foi discutida sobre dois pontos de vista, designadamente, *recuperação de documentos fonte* e *alinhamento de texto para deteção de plágio extrínseco*. Os sistemas para ambas tarefas foram avaliados as suas performance nessa edição através da utilização de um *framework* criado na edição passada. Quanto a tarefa de *alinhamento de texto para deteção de plágio extrínseco análise detalhada do documento* abordou-se na perspetiva de plágio ofuscado em: (i) tradução cíclica (*cyclic translations*), uma nova estratégia de emulação de plágio e (ii) resumo/sumarização (*summaries*), compressão do texto sem alterar o seu sentido original.

No ano de 2014, as duas áreas discutidas na edição anterior (*recuperação de documentos fonte* e *alinhamento de texto para deteção de plágio extrínseco*), foram novamente discutidas [PHG+13, PHB+14]. Desta vez, para deteção de plágio, em que o plagiador tente omitir através da ofuscação, novas métricas foram criadas e automatizadas, sendo mais realistas por terem os cenários do mundo real como campos de treino, avaliação e teste.
Neste mesmo ano, Quoc e Mikolov [LM14] realizaram um estudo no qual fizeram representação distribuída das frases de documentos de texto através da transformação dos parágrafos em vetores (*vector paragraph*) e, dado uma palavra, prediz-se o contexto em que tal palavra ocorre. Este estudo demonstrou um avanço na melhoria da *recuperação de documentos fonte* e da *análise detalhada de documento* (discutido no Capítulo 3, em particularmente na Secção 3.4).

No ano de 2015, Stamatatos *et al.* [SPP+15] apresentam a problemática da **autoria** e propõem soluções através de: (1) *deteção de plágio* (conceito já apresentado) , (2) *identificação do autor* (*author identification*) e (3) *perfil do autor* (*author profiling*).
A *Identificação do autor* diferente da *verificação da autoria* essencialmente no número dos autores. Em *verificação do autor* [SKS07], temos um "único autor" para "verificar" se é ou não o autor de um "texto anónimo". Já em *identificação do autor*, através do "estilo de escrita" e outras métricas, o desafio é determinar a similaridades entre as peculiaridades de cada texto e ser capaz de identificar entre vários autores e diversos textos quem é o autor de um determinado texto entre os diversos textos da coleção. Enquanto que no *perfil do autor*, particularmente para perfis nas redes sociais, os autores são agrupados em classes para serem estudados alguns aspetos social tais como: língua em que partilha, língua nativa, género, idade ou tipo de personalidade afim de serem aplicados em áreas como forense, segurança e marketing [SPP+15]. Diferente de [PSE+09, PBcE+10, PEBc+11, PHG+12, PHG+13, PHB+14, HPS15] o *alinhamento*

de texto para deteção de plágio extrínseco é baseado na revisão dos *corpus* para adequa-los aos tipos de ofuscação. Foram fundamentalmente revisados em: (1) **coleção de *corpus***: reunir textos reutilizados ou plagiados e anota-los como instâncias do mundo real; (2) **geração de *corpus***: dado um par de documentos, gera passagens de reutilização de textos ou plágio entre si [SPP+15].
Ainda neste mesmo ano, [KSKW15] outra abordagem, *Word Mover's Distance (WMD)* que mede quão desiguais são dois documentos de texto pelo mínimo de distância que o aglomerado de palavras do documento de origem A tem com outro aglomerado no documento de destino B. Esta abordagem despertou o nosso interesse de forma particular (discutida nas secções 2.3.2 e 3.4 respetivamente).

Em 2016, Rosso *et al.* [RRP+16, STV+16] abordaram sobre a questão da autoria dividida em três tarefas distintas: (i) *identificação do autor*, (ii) *perfil do autor* e (iii) *author obfuscation evaluation* (avaliação da ofuscação do autor). Cada uma das três áreas teve suas tarefas específicas. O *perfil do autor* por ser semelhante a descrição feita por [SPP+15] abordada acima, a seguir já não falaremos.
- A tarefa *identificação do autor* teve uma atenção especial dos autores e foi sub-classificada em (1) *author clustering* (aglomerado de autor) e (2) *author diarization* (diarização de autor). Dada as dificuldades encontradas pelo facto de que vários apelidos podem pertencer ao mesmo autor, surgiu a necessidade de estudar dois outros cenários derivados de *aglomerado de autor*: $a)$ *Complete author clustering* (aglomeração completa de autores): neste cenário são exigidas análises detalhadas, como o número dos diferentes autores e cada documento terá de ser assinado exatamente por um dos autores. $b)$ *authorship-link ranking* (classificação da ligação da autoria): neste cenário, considera-se a exploração da coleção de documentos como sendo uma tarefa de *recuperação de documentos fonte*.
Quanto ao *diarização de autor*[1], sub-classificada em três grupos distintos, designadamente: (1) *traditional intrinsic plagiarism detection* (*deteção de plágio intrínseco tradicional*), assume-se que um autor principal escreveu pelo menos 70% de um documento, todavia, o desafio é encontrar as partes de texto restantes escritas por um ou vários outros. (2) *diarization with a given number of authors* (*diarização com determinado número de autores*), sabe-se o número de autores que escreveram um determinado documento; o desafio é agrupar cada segmento do texto ao seu autor. (3) *unrestricted diarization* (*diarização sem restrições*), o número de autores não é conhecido e a tarefa passa em encontrar quê autor escreveu determinado segmento de texto e identificar o número exato de autores que escreveram o texto analisado.
- A *avaliação da ofuscação do autor*: é o uso da tecnologia com objetivo de impedir disfarces da autoria de um dado texto. Tanto *author masking* (disfarce do autor) quanto a *avaliação da ofuscação do autor* têm como objetivo de identificar o autor do texto mesmo que haja tentativas obstruir o estilo de escrita deste autor em um determinado texto tornando-o diferente dos outros textos da sua autoria. [RRP+16]
Tschuggnall *et al.* [TSV+17] abordaram a questão da *unsupervised author analisys* (análise de autor sem supervisão) em dois aspetos (1) *style breach detection* (deteção de quebra de estilo) e (2) *aglomerado de autor*. Em [HPS17] discutem a problemática de *ofuscação de autor*. Após uma ofuscação que ultrapassa a capacidade de leitura humana, mas que com base na tecnologia permitiu a reorganização das paráfrases, vinculações textuais, gramática e deteção de estilo.

[1] *author diarization*, tendo um ou vários documentos, identifica as partes dos documentos que foram escritas pelo mesmo autor.

2.2 Métodos de Recuperação de Documentos Fonte

A **recuperação de documentos fonte** consiste numa coleção com vários documentos sobre variados assuntos e, através de determinados procedimentos, obtém-se um grupo restrito de documentos que abordam o mesmo assunto que o documento suspeito [HS11, PHG+12, ASA12, PHG+13, PHB+14, FC16, HPA+17]. A este grupo de documentos resultante, designa-se por **documentos candidatos**, como ilustrado na Figura 2.1. Hagen *et al.* [HS11, HPA+17] descrevem *recuperação de documentos fonte* como sendo a etapa crucial no processo de deteção de plágio extrínseco. Sendo que, quando ela é mal feita, compromete as etapas subsequentes.
Por razões de eficiência computacional vários métodos de recuperação de documentos fonte têm sido estudados para melhoria significativa na redução do espaço de procura de modo a aumentar o grau de similaridade entre documentos candidatos e o documento suspeito, no menor tempo possível.
Em [MSC+13, MYZ13, HEB13, KSKW15, LZM15, FC15, ASI+17] são abordados vários métodos de *recuperação de documentos fonte*, por exemplo em [HEB13] para solucionar o problema, na sua abordagem, recorreu a dois algoritmos, espeficiamente *KP-Miner*, para extração das frases-chave em cada documento e *TextTiling* usado para tokenização do texto em pseudo-frases (segmentos) de tamanhos fixos.
A *recuperação de documentos fonte* quando a coleção tem umas dezenas de documentos ou dois textos é uma a tarefa relativamente menos complexa. O grau de complexidade começa a crescer quando a *recuperação de documentos fonte* envolve documentos suspeitos que tem tentativas de ofuscação de plágio [AIAA15, ASP15], com paráfrases, reordenação de palavras, substituição de palavras pelos seus sinónimos numa clara tentativas de omissão da atitude fraudulenta, em uma coleção contendo centenas de milhares de documentos, tal como na *Web*. Em [FC15, FC16] abordam a *recuperação de documentos fonte* através da indexação do *corpus* fonte no motor de pesquisa e a extração de documentos fonte candidatos por meio de palavras relevantes e características específicas do documento suspeito. Em [FC16], processo de *recuperação de documentos fonte* é particionado em várias fases como: indexação da fonte, formulação da chave de pesquisa e pesquisa e filtragem da fonte.

2.2.1 Formulação da Chave da Pesquisa (**Keyphrase**)

A formulação da chave de pesquisa normalmente ocorre após o pré-processamento do documento suspeito. Várias técnicas de *keyphrase extraction* (extração de frases-chave) são utilizadas e algumas da quais apresentamos de seguida.

Técnicas de extração das palavras-chave **(Keywords)**

Hagen e Stein [HS11], apresentam uma abordagem de formulação da *query* que, com um conjunto de *palavras-chave* W tal como $W = \{w_1, ..., w_n\}$ em que w_i é o *i-ésimo* elemento do conjunto W. Deste conjunto, deriva um subconjunto oriundo das *palavras-chave* do documento suspeito designado por *query* ϱ formado por $\varrho = \{q_1, ..., q_m\}$ em que $q_i \subseteq W$. As *queries* são padronizadas a fim de corresponderem à capacidade limite admitido pelos motores de pesquisa. Se se encaixarem nos padrões, são enviadas ao *motor de pesquisa*, caso contrário são descartadas. Para evitar a redundância nas *queries* e nos resultados, os termos que estiverem em uma *query*, não podem figurar em outra. E, para proteger a capacidade de funcionamento do sistema, cada *query* deve devolver um número limite de resultados.

Em [Mal16] a formação das *queries* é feita com base nas características de cada segmento. Por exemplo, as *queries* são formadas pela sequência concatenada dos *tokens* com a finalidade de formar uma sequencia disjunta das *10-gram*. As três primeiras *10-gram* de cada parágrafo são submetidas ao *motor de pesquisa* e os três melhores resultados por *query*, são devolvidos pelo *motor de pesquisa*.
Potthast *et al.* [PHG+12, LHS+12, PHG+13, PHB+14, SB14, HPS15, HPA+17] abordam diversas formas de formação da *query*. Em [HPA+17] apresentam dois níveis na formulação da *frases-chave*, (1) ao nível do segmento suspeito e (2) ao nível do documento suspeito. Ao nível do segmento são feitas duas *queries* por segmento. As dez melhores *palavras-chave* são divididas em dois grupos de *5-gram* cada segundo a sua classificação. O primeiro *5-gram* para primeira *query* e o próximo *5-gram* para a segunda *query*. Ao nível do documento formulam-se igualmente duas *queries*. Em [LHS+12] a formação da *query* é feita com base nos melhores resultados do *Term Frequency-Inverse Document Frequency (TF-IDF)* (frequência do termo e frequência documental inversa) de cada termo em um determinado parágrafo do documento suspeito. Hagen e Stein [HS11] afirmam que, a adição de vocábulos longos provenientes do documento suspeito na formulação de *keyphrases* aumentam a garantia de que os resultados devolvidos sejam de facto documentos candidatos que abordam o mesmo assunto ou tópicos altamente similares.

O Algoritmo de *K-Miner* [EBR10], a biblioteca *NLTK*[2], o indexador *Lucene* [Mal16, HPS15, HPA+17], são alguns dos recursos utilizadas para extração das *palavras-chave e frases-chave*, quer ao nível do segmento quer ao nível do documento, na extração dos vocábulos longos [HS11], e as palavras que mais se repetem no documento (*TF-IDF*).
Além dos recursos mencionados acima, utiliza-se ainda *Part-Of-Speech (POS)* para extração ou identificação de termos específicos (*i.e.*, adjetivos, substantivos, entidades, etc.) [HPA+17].
Segundo Florescu e Jin [FJ18] a *extração de frases-chave* tem vindo a ser proposta em duas linhas de pesquisa, a supervisionada e a não supervisionada. Na abordagem supervisionada, as orações são rotuladas com um conjunto de caraterísticas, como as suas *TF-IDF*, posições no documento, ou rótulos de *POS*, algoritmos de aprendizagem máquina são treinados para classificar as orações como positivas, (*i.e., keyphrases*) ou negativa (*i.e., non-keyphrases*). Na abordagem não supervisionada, segundo Bennani-Smires *et al.* [BSMJ+18], existe restrições de informação de duas maneiras, $a)$ confinar informações estatística interna i.e. frequência documental inversa das palavras e $b)$ utilização de informações extraídas somente no documento em análise.
A eficácia na extração das *frases-chave* no documento suspeito, determina a eficácia na recuperação de documentos fonte.

2.2.2 Expansão da *Query*

A expansão da *query* é um processo de adição de termos relevantes na *query* para melhorar a performance da recuperação [ZC16, NSC17]. Para palavras relevantes são encontradas sinónimos incluídos na *chave de pesquisa*. Esta técnica pode ser utilizada em análises de documentos local ou global. Em [PHG+13] refere-se que, tendo um conjunto de *palavras-chave* extraídas dos segmentos de um determinado documento, a *query* é formulada de acordo as restrições do *motor de pesquisa*. São várias as políticas de restrições de números de termos que uma *query* pode ter. Pelo que os autores [PHG+13, SB14, HPS15, ZC16, NSC17, HPA+17] abordam, podemos concluir que a expansão da *query* se dá em quatro fases distintas: (i) Pré-processamento, (ii)

[2] http://www.nltk.org/

formulação da *query*, (iii) recuperação e (iv) agrupamento/mistura dos resultados.
Nawab *et al*. [NSC17], utilizam o método de Recuperação de Informação (RI) baseado na expansão da *query* quando há ofuscação do plágio.

2.2.3 Indexação Fonte

O processo de indexação ocorre após pré-processamento do documento suspeito (remoção de imagens, tabelas, gráficos, remoção das *stopWords, stemming*), transformação das letras em minúsculas, entre outras [PHG+12, SB14, FC15, ASI+17]. De seguida, cada documento é segmentado e cada segmento é rotulado e registado no indexador.
À prior, todos os documentos deverão estar indexados no *corpus* onde posteriormente far-se-á a pesquisa [PHG+12].
Uma das ferramentas mais utilizados atualmente para indexação por conta do seu bom resultado é *Apache Lucene*[3].

2.2.4 Pesquisa e Filtragem da Fonte

Para a pesquisa e filtragem da fonte, precisamos considerar um conjunto de *queries* e um conjunto de documentos obtidos resultante da submissão das *queries* ao *motor de pesquisa*. Na filtragem, todos os documentos retornados são analisados e descartam-se aqueles documentos que não se mostram pormenores, reduzindo assim o número dos documentos candidatos. Todos os documentos não descartados, transitam para etapa seguinte [SB14].

2.2.5 Seleção dos Documentos Candidato

As *palavras-chave* extraídas dos segmentos do documento suspeito, após a formação das *queries* que, por sua vez são enviadas ao *motor de pesquisa*, são as responsáveis pelos documentos devolvidos.
Hagen e Stein [HS11] afirmam que o objetivo nas abordagens de seleção de documentos candidatos não é somente encontrar cópias exatas (*i.e., one-to-one*) mas também paráfrase [CDB07]. Em [ASA12] é nos apresentado *IR Method* (método de RI), *Clustering Techniques* (Técnicas de aglomeração) e *Cross-Lingual Retrieval Models* (Modelos de recuperação com língua cruzada) como formas de seleção de documentos candidatos. Leilei *et al*. [LHS+12] apresentam uma abordagem de *recuperação de documentos fonte* baseado em TF-IDF, para extração das *palavras-chave* do documento suspeito. Para qualquer documento suspeito, as *queries* são provenientes da melhor classificação dos termos do grupo das *queries* que respeitam os limites mínimos e máximos de caracteres, ordenado por TF-IDF, do maior para o menor em cada parágrafo do documento suspeito.
Nawab *et al*. [NSC17] apresentam uma abordagem de *recuperação de documentos fonte* utilizando o método *Kullback-Leibler Symmetric Distance*. Os documentos são modelados como distribuição probabilística e comparados utilizando KL_δ, posteriormente são convertidos em distribuição probabilística com a remoção das *stopWords*, *stemming* e em seguida cálculo do peso TF-IDF para as palavras restantes. O cálculo do *kullback-leibler symmetric distance* é realizado segundo a Equação 2.1.

[3]https://lucene.apache.org/

$$KL_\delta(P_d||Q_s) = \sum_{x \in X}(P_d(x) - Q_s(x))log\frac{P_d(x)}{Q_s(x)} \quad \textbf{(2.1)}$$

O P_d representa a distribuição probabilística gerada por d, em que d representa o documento na coleção de referência e Q_s é a distribuição equivalente para s, o documento suspeito. X é o vetor formado pelo conjunto de palavras do segmento do documento suspeito e, x cada uma destas palavras. Os autores [NSC17] afirmam que, com a aplicação do *Kullback-Leibler Symmetric Distance* melhorou a redução substancial do tempo de execução.
Cada fase da etapa de *recuperação de documentos fonte*, tem implicações nas outras fases subsequentes.

2.3 Método de Análise Detalhada

A análise detalhada (*detailed analysis* ou ainda *text alignment*) é a etapa de deteção de plágio propriamente dita. Os *métodos de análise detalhada de documentos* são normalmente os mesmos utilizados pela primeira etapa desta tarefa, ou seja, *recuperação fonte de documentos candidatos*. Nesta Secção serão abordados de forma síntese alguns dos *métodos de análise detalhada* existentes na literatura.
A DAP (intrínseco e extrínseco) é uma tarefa que baseia-se fundamentalmente na matemática e faz recurso a tecnologia aplicando um conjunto de princípios e equações por forma a combater atitudes fraudulentas existentes na literatura, com particular realce na *Web*. Sendo assim, agrupamos estes métodos em duas categorias, descritas a seguir.

2.3.1 Alguns Métodos de Análise Detalhada mais Utilizados

De algum tempo a esta parte, os métodos de analise de plágio têm sido estudados de forma a melhorar a sua eficiência sem perder a eficácia a medida que novas práticas de plágio são utilizadas. Sabendo que alguns métodos de análise detalhada consideram a deteção de plágio focando em **cópias exata**, Alzahran *et al.* [ASA12] baseados em características textuais (i.e., léxicas, sintáticas, semânticas, estruturais) para deteção de plágio, agruparam os métodos em função a duas taxonomias de plágio, as *cópia exata, aproximada, e cópia modificada*, chamou de **plágio literal** e as *manipulação de texto (paráfrases e sumarização), tradução (automática ou manual), adoção de ideia (baseada no significado, baseada na importância e baseado no contexto)*, chamou-os de **plágio inteligente**. Nesta pesquisa identificaram alguns métodos de deteção de plágio com cópia exatas, designadamente: - Bag-Of-Words (BOW), que segundo Quoc *et al.* [LM14, MSC+13, AlAA15, ASI+17] corroboram e dizendo que, duas frases podem ser ditas em contextos diferentes e utilizado *BOW* serem identificadas como plágio, quando não são, por exemplo:

> *Cai neve na serra da estrela*
>
> *Caí na neve da serra da estrela*

- Outros métodos que os autores agrupam nesta categoria é o Vector Space Model (VSM) original e o Latent Semantic Indexing (LSI). Para eles, ambos os métodos consideram a similaridade do documento como um todo, o que não ajuda para deteção de segmentos ofuscados [ASA12].

Os métodos de deteção de plágio com **cópias modificadas** estão entre os que figuram na tabela IV de [ASA12], mormente *Semantic-Based Methods*, *Fuzzy-Based Methods*. Na sua pesquisa estes métodos detetaram o tipos plágio modificados por paráfrases. Referem que, algumas das técnicas que tornam os métodos de deteção de plágio adequados para *deteção de plágios inteligente* são: *n-grams* (com as suas variações), *TF-IDF, POS, stemmer, lemmatizer* entre outros.
Em [AIAA15, ASI$^+$17] é apresentado uma abordagem que, quando as frases têm o mesmo BOW, combina as características semântica e sintática da informação contida no texto recorrem à análise da estrutura sintática utilizando POS para determinar a informação semântica entre as duas frases. Utilizam a *similaridade do cosseno* para identificação do plágio. Em [LHS$^+$12] *similaridade do cosseno* é também o método usado para a análise detalhada com a particularidade de ocorrer em três fases:
Fase 1: recuperação de frases similares, ambos os documentos (suspeito e fonte) são divididos em frases e todas as frases do documento suspeito são comparadas com as frases fonte. A comparação considera um *limiar* de decisão expresso na Equação 2.2 por t_1.

$$sim(S,R) = \frac{\sum_{k=1}^{n} w_{Sk} \times w_{Rk}}{\sqrt{(\sum_{k=1}^{n} w_{Sk^2})(\sum_{k=1}^{n} w_{Rk^2})}} > t_1 \quad \textbf{(2.2)}$$

Assim, $sim(S,R)$ é o grau de similaridade entre S e R, θ é o ângulo do documento, w_{Sk} e w_{Rk} são os pesos de S e R respetivamente, t_1 é o *limiar* de decisão da similaridade entre as frases.

Fase 2: frases candidatas a plágio, os autores calculam o grau da semelhança da estrutura das frases candidatas a plágio utilizando a Equação 2.3, designada por *Overlapping Measure Model*.

$$T = \frac{2 * \sum_{t \in I_S \cap I_R} Min(N_{I_S}(t), N_{I_R}(t))}{|I_s| + |I_R|} > t2 \quad \textbf{(2.3)}$$

Onde $N_{I_S(t)}$ e $N_{I_R(t)}$ são o número de termos comum das frases suspeita e fonte, $Min(N_{I_S}(t), N_{I_R}(t))$ é o valor mínimo do somatório dos termos na frase suspeita com os termos na frase de fonte, I_S e I_R são o tamanho das frases suspeita e fonte respetivamente contabilizando o número de palavras que as compõem e, $t2$ é o *limiar* de decisão.

Fase 3: par candidato resultante, após as duas fases anteriores, eventualmente algumas frases igualmente plagiadas podem ser mal articuladas e perder-se a possibilidade de detetar mais um ou outro segmento plagiado. Nesta terceira fase, faz-se a nova procura para identificar estas situações caso existam.
Em [PHG$^+$12, PHG$^+$13, PHB$^+$14, SPSG14, FC16] apresentam semelhanças na forma de lidar com a **detailed analysis**, através da implementação de estratégias baseadas em três categorias de algoritmos; a saber: (1) *seeding* (sementes), (2) *extension* (extensão), (3) *filtering* (filtragem). Baseado em heurísticas, a *seeding* ou *seed*, também chamada de *matches* (correspondências), entre os documentos (fonte e suspeito) são identificados os pares que têm uma correspondência exata. Os que têm correspondência aproximada que pode se subentender por ofuscação, reordenação das palavras ou substituição pelos seus sinónimos, passam por um processo de criação de correspondência produzindo mudanças básicas entre ambos documentos (caso as mesmas se justifiquem), para identificar plágio, não só nas frases ou nos segmentos, mas no documento como um todo. Para tal, são usadodos n-grams em diversas variações *i.e., 3-grams, 4-gram,*

5-gram, 8-gram e *1-skip-3-grams*.
O *TF-IDF* e o VSM são adotados por Sanchez-Perez *et al*. [SPSG14] para aferir o grau de similaridade entre duas frases. É considerado um caso de plágio, sempre que a frase suspeita tenha palavras comuns com a frase fonte cujo valor percentual ultrapassa um determinado *limiar*. Assim, são "vectorizadas" as frases em vez dos documentos, adaptando-se a *IDF* para *inverse sentence frequency* (*ISF frequência inversa da frase*).

$$tf(t,s) = f(t,s) \quad (2.4)$$

$$isf(t,D) = log\frac{|D|}{|\{s \in D : t \in s\}|} \quad (2.5)$$

$$w(t,s) = tf(t,s)xisf(t,D) \quad (2.6)$$

Onde a frequência do termo $tf(t,s)$, utiliza o número de ocorrência $f(t,s)$ do termo *t* na frase; D é o conjunto de todas as frases em ambos os documentos dados, e $w(t,s)$ é a *t-ésima* coordenada da frase s na representação de VSM. Sendo assim, o par de frases suspeitas $susp_i$ e fonte src_i provenientes dos documentos suspeito e fonte pertencem em s obedecendo as seguintes condições:

$$cos(susp_i, src_j) = \frac{susp_i.src_j}{|susp_i|.|src_j|} \geq th1 \quad (2.7)$$

$$Dice(susp_i, src_j) = \frac{2|\delta(susp_i).\delta(src_j)|}{|\delta(susp_i)|^2 + |\delta(src_j)|^2} \geq th2 \quad (2.8)$$

Sendo as frases suspeita e fonte representadas como vetores, cos a medida do cosseno, *Dice* é o coeficiente de *Dice*, $|.|$ é a distância Euclidiana, e $th1$ e $th2$ são os *limiares* de similaridade e devem ser diferentes de zero.
Na categoria *extension*, as sementes identificadas como correspondentes entre os documentos suspeito e fonte, são estendidas em textos alinhados por segmentos dos dois documentos com o comprimento máximo, o qual são reportados como deteção de plágio. Ainda segundo estes autores, faz-se a correspondência da semente para identificação de plágio não só nas frases dos segmentos mas no documento completo. E, a regra exata da heurística depende da semente utilizada. Neste categoria, (*filtering* [SPSG14]), removem-se algumas "más" deteções de casos de plágio em duas fases: primeiro, para *remoção da sobreposição dos fragmentos*: quando uma frase é repetida várias vezes no documento suspeito e é utilizada uma única vez no documento fonte, é detetada como sendo vários casos de plágio quando, só foi plagiada uma vez sendo, um único caso de plágio. Segundo, *remoção das sobreposições dos pequenos fragmentos*: remoção de todos segmentos que são extremamente pequenos. Não existe um limite claro de caracteres e/ou palavras a ignorar [PHG+12, PHG+13, PHB+14].
Outros métodos como Jaccard Coefficient for Keywords Similarity [NSNW13, ASI+17], análise detalhada com índice invertido [Sta09], *KP-Miner* [EBR10], *Levensthein Distance Algorithm* [CDB07, NB17], *Word Simple N-gram Overlap, Exclusive LCP N-gram, The BLEU Metric* [CDB07], *Latent Dirichlet Allocation* e *Kullback-Leibler Symmetric Distance* [NSC17] também são utilizados.

2.3.2 Word2Vec

A necessidade da análise detalhada à medida da sofisticação das práticas do plágio, é cada vez mais urgente e necessária. O recurso aos denominados *embeddings* mostram-se promissores na DAP. O mais conhecido método de *embedding* é o *word2vec*. Desenvolvida por Miklov *et al.* [MCCD13], em que uma rede é treinada com um *corpus*, onde cada palavra do *corpus* tem o seu significado representado através de um *vetor semântico*. Sendo assim, o *word2vec* é uma forma de representação distribuída de uma coleção de palavras num espaço vetorial utilizado em PLN para identificar relação semântica entre as palavras [MSC+13].
Tipicamente, o *word2vec*, é baseado em duas arquiteturas e/ou algoritmos *e.g., CBOW*: prediz a palavra e anuncia-se o contexto. E, *Skip-gram model*: prediz-se o contexto e ele anuncia a palavra [MCCD13, MYZ13, Dan17].
A identificação das características textuais como pré-requisito na tarefa de deteção de plágio, é efetuada através do conhecimento das palavras a volta da que está a ser analisada. No nosso entender, esta é a melhor forma de conhecer o contexto do texto[4] O *word2vec* dá resposta a este problema pelo facto de cada palavra ter o seu *vetor semântico* (significado da palavra).
Vários pesquisadores têm implementado *métodos de deteção de plágio* usando *word2vec*. Em [KSKW15] a comparação do documento no seu todo permite criar o grau de similaridade entre os documentos, transportando as palavras de um documento para o outro, (detalhado na Subsecção 3.4). Em [HGK+16] tomam os resultados de [KSKW15] faz-se uma transformação linear da representação das palavras para detetar os *matches* pelos significados das palavras através das suas codificações. Estes dados são obtidos de duas maneiras: primeiro, após a transformação do *Word Embedding*, toma-se a representação que é adaptada em *transformação linear*; segundo, apresentam várias formas de classificação da palavra partindo do pressuposto que o valor de uma palavra em uma coleção difere em relação ao contexto. Huang *et al.* [HGK+16] apresentam uma métrica evolutiva de **WMD** supervisionada (ver Huang *et al.*, [HGK+16]).

Normalmente o *embedding* é formado por uma matriz $\mathbf{X} \in \mathbb{R}^{d \times n}$ para um tamanho finito de vocabulário de n palavra. A i-éssima coluna, $\mathbf{x_i} \in \mathbb{R}^d$, representa o *embedding* da i-éssima palavra da coluna no documento d [KSKW15, Dan17].

2.4 Pós-processamento Baseado no Conhecimento

O pós-processamento baseado no conhecimento, é uma etapa em que os pares de segmentos oriundos de ambos os documentos (suspeito e fonte) são analisados em relação a informação referida e visualizada para perícia necessária [HPA+17]. Após a apresentação dos resultados num formato *legível*, é quando se determina efetivamente quê segmento do documento suspeito (ou parte do segmento), é plágio e em quê segmento do documento fonte (ou em quê parte do segmento) foi plagiado [ASA12, ES16]. Em alguns casos, a informação contida nos documento candidato ou em seus segmentos, coincidem simplesmente com a informação do documento suspeito ou também em seus segmentos. Nestas situações, a possibilidade de caso de plágio é descartada nesta etapa [MSY14, ES16]. Segundo Ehsana e Shakery [ES16], em alguns segmentos apresentados nesta etapa podem não ser plagiados. Esta é, portanto, a etapa em que é verificada toda a sobreposição dos pares dos segmentos dos documentos e, após o *output* ficará

[4] contexto do texto: é definido pelo ambiente imediato em que determinado texto está sendo produzido [Mey87].

conhecido o resultado do detetor de plágio [AIAA15, ASI+17].

2.5 Sumário

As três principais etapas da deteção de plágio externo, estão intrinsecamente ligadas visto que, se não tivermos os documentos candidatos certos, não teremos sucesso na *análise detalhada* e se houver falhas significativas na *análise detalhada*, comprometemos os resultados do *pós-processamento baseado no conhecimento*. Sendo assim, podemos afirmar que a ineficácia de uma etapa, condiciona as seguintes.
Este assunto tem sido investigado pela comunidade de pesquisadores de todo mundo, tendo em conta a sua relevância e atualidade. Em diversas conferências (PAN,Conference and Labs of the Evaluation Forum (CLEF) etc.) são anualmente apresentados novos resultados no domínio de DAP, e aqui ficou um resumo das principais abordagens encontradas.

Capítulo 3

Análise Detalhada de Plágio

No presente Capítulo, apresentámos as experiências das implementações feitas com três algoritmos distintos de *análise detalhada de documento*. As nossas experiências de implementação foram com os algoritmos de *pesquisa linear, análise recursiva* da similaridade de um documento fonte em relação à uma frase suspeita; em duas abordagens (divisão recursiva binária e ternária respetivamente). Nessas experiências utilizamos a *similaridade de Jaccard* (nas duas abordagens referidas anteriormente) e, os *embeddings* de *word2Vec* numa nova abordagem. Todas experiências foram no sentido de identificar métodos e técnicas de *análise detalhada* para deteção de pequenos e grandes segmentos de textos extraídos de documentos de grande dimensão.
Como veremos na Secção 3.1, apresentamos de forma sintética, alguns aspetos relevantes dos métodos e técnicas implementados nesta dissertação.

3.1 Generalidade sobre os Métodos e Técnicas de Análise Detalhada Implementados

Num universo de vários métodos de *análise detalhada de documento* (também designada por alinhamento de texto, do inglês, *text alignment*), apresentados na literatura, alguns dos quais mencionados no Capítulo 2, na Secção 2.3, referidos em [LMD01, BCRB09, PEBc+11, TP10, PHG+12, MSC+13, MCCD13, HEB13, SPSG14, RRP+15, SPP+15, AIAA15, ASI+17, Dan17, HPA+17, BSMJ+18], implementámos três algoritmos de Deteção Automática de Plágio (DAP) extrínseco (*i.e.*, copiar segmento de um documento e utilizar no outro documento sem citar fonte).
As nossas medições e experiências foram realizados com *corpus* PAN12 [PHG+12]. Começámos a realizar as nossas experiências com a implementação de algoritmos de *pesquisa linear*, em que se compara palavra a palavra entre uma frase suspeita e outra frase fonte resultante da segmentação de um documento fonte que produz uma coleção de frases fonte. A seguir, implementámos o Algoritmo de *Divisão recursiva* [SPSG14] *do documento fonte em grandes blocos e pesquisa em cada bloco*. Variámos este Algoritmo em duas abordagens principais: primeiro, com *divisão do documento fonte em dois grandes blocos*; e segundo, com a *divisão do documento fonte em três grandes blocos*. Em cada uma das abordagens realizámos as experiências de deteções de plágio com duas equações distintas. Com a Equação 3.1 que designamos por *métrica da similaridade entre os vocabulários* [LMD01] e a Equação 3.2 *métrica da similaridade de Jaccard* [NSNW13].
Por último, implementámos o *word embeddings* (*word2Vec*) [MSC+13, MCCD13, MYZ13, Dan17] em que a frase suspeita é o *vetor semântico* que define o contexto[1], e o tamanho de cada segmento fonte a ser comparado com o segmento suspeito. Este tamanho é fixo quanto ao comprimento e é variável quanto a sua semântica em cada iteração (abordado na Secção 3.4).
A primeira experiência das implementações é apresentada na Secção 3.2, a seguir.

[1] âmbito em que uma frase fonte é utilizada [Mey87]

3.2 Pesquisa Linear

Como anunciamos na Secção 3.1, para as situações em que o "plagiador" ao praticar o plágio não se preocupa com a ofuscação, e partindo do pressuposto que, por mais que ele enverede pela prática de ofuscação com reordenação das palavras, substituição por sinónimos (i.e., paráfrases) não conseguirá eliminar o léxico do documento fonte na totalidade [ASI+17]. Com base nisto, implementámos o Algoritmo 1 de *pesquisa linear* (sequencial), que a seguir apresentamos:

Algorithm 1 Pesquisa Linear

1: **Input:** δ, d
2: $d = [s_1, ..., s_n]$
3: $\delta = [\delta_1, ..., \delta_k]$
4: $casos \leftarrow \emptyset$
5: **for** $s \in d$ **do**
6: $i \leftarrow \emptyset$
7: **while** i < $min(|\delta|, |s|) \wedge \delta[i] = s[i]$ **do**
8: $i \leftarrow i + 1$
9: endwhile
10: $Escrever(casos)$
11: endfor

O documento suspeito δ é segmentado numa lista de frases, e cada frase é transformada numa lista de palavras. Este processo é igualmente aplicado ao documento fonte d. A variável *Casos*, recebe as sequências de correspondências encontradas (*i.e., matches*) entre δ e s. Estes *matches* são a correspondência exata das palavras encontradas na mesma sequência (em δ e s), ilustrado na Figura 3.1.
Na linha sete, a função *min* identifica os tamanhos de δ e s e *i* identifica a posição da frase δ no documento d, onde δ foi extraído.

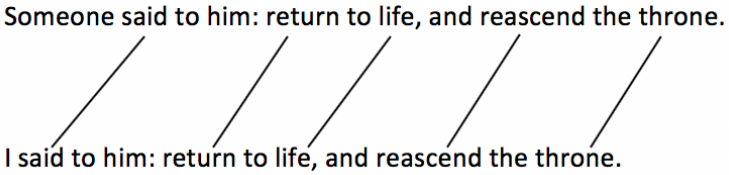

Figura 3.1: Matches: pesquisa linear

A figura acima, ilustra um caso detetado utilizando o Algoritmo de 1. A primeira frase é a δ e a outra é a s. As palavras sem ligação com o seu par na outra frase foram ignoradas por serem palavras frequentes e menos informativa (*stopWords*). Neste exemplo, temos um *matche* com cinco sequências de palavras.
Para identificação das características textuais, entre frases ou textos, é necessário saber as palavras comum entre ambos. Por isso, na Secção 3.3 apresentamos um algoritmo para criação de vocabulário.

3.3 Análise Recursiva da Similaridade em Blocos de Texto

A pesquisa sequencial em documentos de grande dimensão, é uma tarefa fastidiosa e exaustiva o que consome muitos recursos computacionais assim como tempo. Afim de otimizarmos estes

custos, por um lado e por outro, aumentar a possibilidade encontrar situações de passagens parafraseadas, recorremos à segmentação documento suspeito δ em parágrafos [HEB13, PHB+14] e do documento fonte S em blocos. Analisamos a similaridade dos blocos (resultante da divisão do documento fonte em grandes blocos constituídos por parágrafos), através dos seus vocabulários em comparação com o vocabulário formado com as palavras do parágrafo suspeito. Este processo, ocorre segundo a ilustração presente no Algoritmo 2, a seguir.

Algorithm 2 Criação de Vocabulário

```
1: Input: Text
2: Vocab ← ∅
3: for w ∈ Text do
4:    if w ∉ StopWords then
5:       w_s ← menor(sinonimo(w))
6:       if w_s ∉ Vocab then
7:          Vocab ← Vocab ∪ {w}
8:       endif
9:    endif
10:   return Vocab
11: endfor
```

Para cada segmento do documento criamos o seu vocabulário e depois, comparamos o vocabulário suspeito com todos os vocabulários de todos os segmentos fonte.
Da linha quatro a sete verificamos se a palavra w do documento $Text$ é *StopWords*. Se não for, inserimos esta palavra no vocabulário *Vocab*. Após a primeira inserção de uma palavra no vocabulário *Vocab*, antes de uma próxima inserção verificamos se a palavra já existe ou se existe um sinónimo desta palavra. Caso a palavra já exista, não é adicionada e se seu sinónimo existir, procuramos o menor *sinónimo alfabeticamente* desta palavra (este processo é detalhado em frente). É este sinónimo que é colocado no vocabulário *Vocab*. Se a palavra não existe no vocabulário e nem o seu sinónimo, ela é adicionada. A operação continua pelas restantes palavras do segmento do texto até ao fim.

A Tabela 3.1 ilustra um exemplo da escolha do menor sinónimo da palavra *love*. Neste exemplo, de acordo a ordem das palavras na referida tabela, o menor sinónimo alfabético da palavra *love* é a palavra *benevolence*. Embora as oito restantes palavras sejam sinónimas de *love*, a nossa implementação explicada na linha cinco do Algoritmo 2, adiciona no vocabulário a palavra *benevolence*.

Tabela 3.1: Sinónimo

N/O	Sinónimo
1	benevolence
2	charity
3	devotion
4	endearment
5	fondness
6	goodwill
7	kindliness
8	kindness
9	tenderness

A seguir é apresentado na Subsecção 3.3.1 a utilização do vocabulário na *métrica da similaridade entre os vocabulários*, com a Equação 3.1.

3.3.1 Métrica da Similaridade Entre os Vocabulários

Após a criação dos vocabulários, utilizamos os mesmos afim de medir a similaridade [LMD01, CDB07] entre cada vocabulário suspeito *Vocab(δ)* com todos os vocabulários fonte *Vocab(s)*, um por cada vez. As medições foram essencialmente, o tempo para detetar frases plagiadas (com e sem ofuscação), a quantidade de casos de plágio efetivamente detetados e os casos de plágios não detetados e por último, os segmentos que não eram plagiados e eventualmente foram detetados como sendo plágio. Os detalhes destas medições e das experiências realizadas, são explicados no Capítulo 4.
Ainda sobre a similaridade, definimos um limiar (*i.e., threshold*, limite mínimo para determinar se uma frase δ foi ou não plágio em *s*), inspirados na recursividade abordada em [SPSG14] recorremos a Equação 3.1 [LMD01] para implementar a nossa abordagem.

$$Sim(Vocab(\delta), Vocab(s)) = \frac{|Vocab(\delta) \bigcap Vocab(s)|}{|Vocab(\delta)|} \quad (3.1)$$

Esta função calcula uma similaridade relativa entre dois vocabulários.
Como o nosso objetivo era *criar uma abordagem que nos permitisse identificar pequenos e grandes segmentos de textos extraídos de um documento de grande dimensão*, a métrica 3.1, podia nos permitir alcançar tal desiderato.
Dado que, uma frase δ que eventualmente foi extraída de um documente d ao fazermos a intercessão entre *Vocab*(δ) e *Vocab*(s); estamos a "misturar" uma quantidade de palavras significativa, ainda que essa quantidade seja pequena. Esta ação faz com que se conserva o que existe em comum entre os dois vocabulários. O facto de ser dividida por *Vocab*(δ) tecnicamente aumenta a possibilidade de encontrar grande parte dela em *Vocab*(s).
De lembrar, não precisamos ter o mesmo léxico para que duas ou mais palavras sejam consideradas "iguais", basta serem sinónimas.

3.3.2 Similaridade de Jaccard

Nesta experiência, fizemos algumas modificações no Algoritmo 3 e calculamos a similaridade de Jaccard. Esta similaridade se fundamenta essencialmente na Equação 3.2, apresentada a seguir:

$$Jaccard(Vocab(\delta), Vocab(s)) = \frac{|Vocab(\delta) \bigcap Vocab(s)|}{|Vocab(\delta) \bigcup Vocab(s)|} \quad (3.2)$$

O que acontece com a Equação 3.2 na parte do denominador é semelhante à Equação 3.1, a diferença consiste em, ao dividir um conjunto menor por outro que pode ser muito maior, a possibilidade de encontrar *Vocab*(δ) no conjunto resultante (união de *Vocab*(δ) e *Vocab*(s)) é menor.

3.3.3 Abordagens de Similaridade Recursivas Adotadas

Nesta Secção, apresentamos duas abordagem de pesquisa recursiva de vocabulários dos blocos de texto para identificar pequenos e grandes segmentos de textos que eventualmente tenham sido plagiados.

Baseando-nos nos argumentos de Cedeño *et.al* [BCRB09] que, nem sempre é fácil analisar um documento inteiro. Assim sendo, pensamos em duas possibilidade para divisão do documento fonte. Na primeira abordagem, dividimos o documento fonte em duas partes e analisamos ambas as partes. Definimos um *threshold* de similaridade. A Figura 3.2 mostra casos em que o documento fonte é dividido em dois blocos. As regiões do documento fonte pintadas a amarelo e vermelho representam a zona plagiada. Na imagem C da referida figura, a divisão ocorre separando a zona plagiada do documento fonte. Uma parte plagiada vai para o bloco C_1 e outra para o bloco C_2. Isto pode diminuir a possibilidade de identificação de casos de plágio na totalidade e pode comprometer os resultados esperados. Porque ao pesquisar o plágio, encontraremos uma parte do segmento plagiado. Esta parte *plagiada* pode estar no bloco C_1 ou no bloco C_2. Nunca estará nos dois blocos simultaneamente. Isto nos trará como resultado da deteção parte do segmento plagiado ou a situação de plágio pode passar sem ser detetada se o segmento separado for significativamente menor.

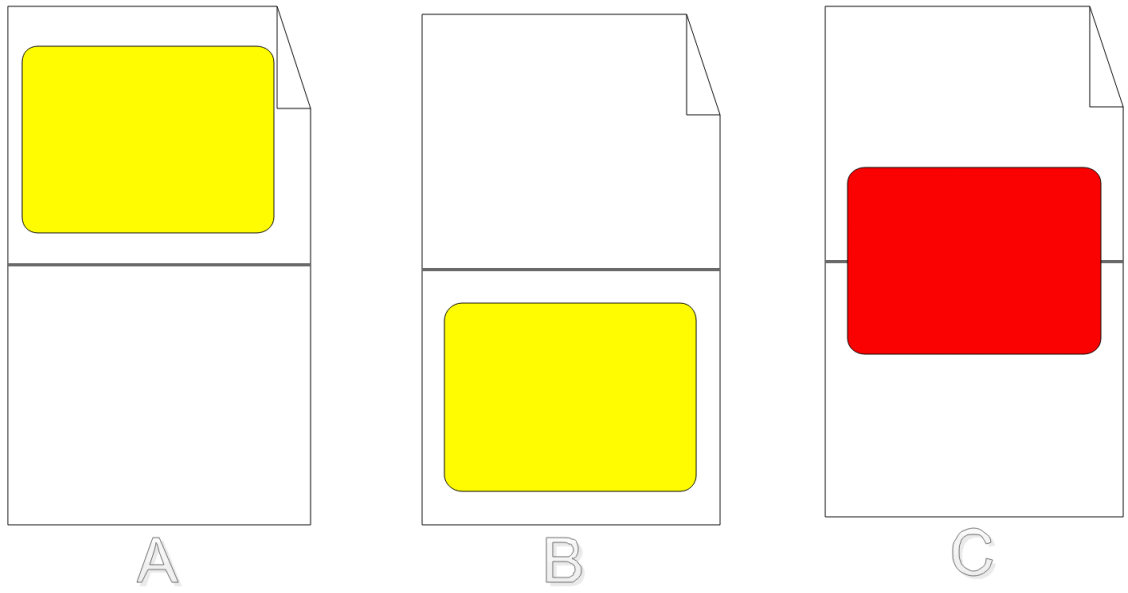

Figura 3.2: Divisão recursiva binária do documento fonte

Este problema foi resolvido com a implementação da segunda abordagem apresentada na Figura 3.3. Nesta abordagem, o documento fonte é segmentado em três blocos, *i.e.*, $D = D_1 \oplus D_2 \oplus D_3$. Quando a zona plagiada também é separada, a pesquisa é feita nos dois blocos que têm partes da zona plagiada. As imagens D e E da Figura 3.3, mostram dois exemplos em que a zona de plágio do documento fonte é separada entre dois blocos. Em D separada nos blocos D_1 e D_2 e em E é separada entre os blocos E_2 e E_3.
Quando a segmentação separa a zona plagiada em dois blocos distintos, fizemos a união dos dois blocos (*i.e.*, $D_1 \cup D_2$), e realizámos a pesquisa nos dois blocos (D_1 e D_2, assim como em E_2 e E_3), através da formação de um vocabulário conjunto. *i.e.*, $Vocab(D_1) \cup Vocab(D_2)$ e $Vocab(E_2) \cup Vocab(E_3)$. A pesquisa é feita no vocabulário resultante (por exemplo, da união de $Vocab(D_1)$ com $Vocab(D_2)$). Desta forma, elimina-se a possibilidade de perder qualquer parte da zona plagiada.

Como trata-se de documentos de texto enorme, na primeira segmentação do documento é possí-

vel que não se encontre ainda o *segmento plagiado*. Para estes casos é aplicada a recursividade na divisão do documento fonte de forma ternária (três partes). Na imagem F da Figura 3.3 a identificação do *limiar* definido é confirmada na primeira segmentação do bloco F_3. Isto significa que o bloco é formado por três sub-blocos *i.e.*, $F_3 = F_{3.1} \oplus F_{3.2} \oplus F_{3.3}$. E na segmentação deste sub-bloco, a zona plagiada foi localizada entre $F_{3.2} \cup F_{3.3}$. É com as palavras existentes nestes dois segmentos que formamos o vocabulário união; isto é: $Vocab(F_{3.2}) \cup Vocab(D_{3.3})$ e faz-se a pesquisa no vocabulário união (entre $Vocab(F_{3.2})$ e $Vocab(D_{3.3})$).
Portanto o problema identificado na primeira abordagem foi resolvido na segunda abordagem com a deteção dos casos de plágio na totalidade sendo qual fosse a sua zona de extração no documento fonte.

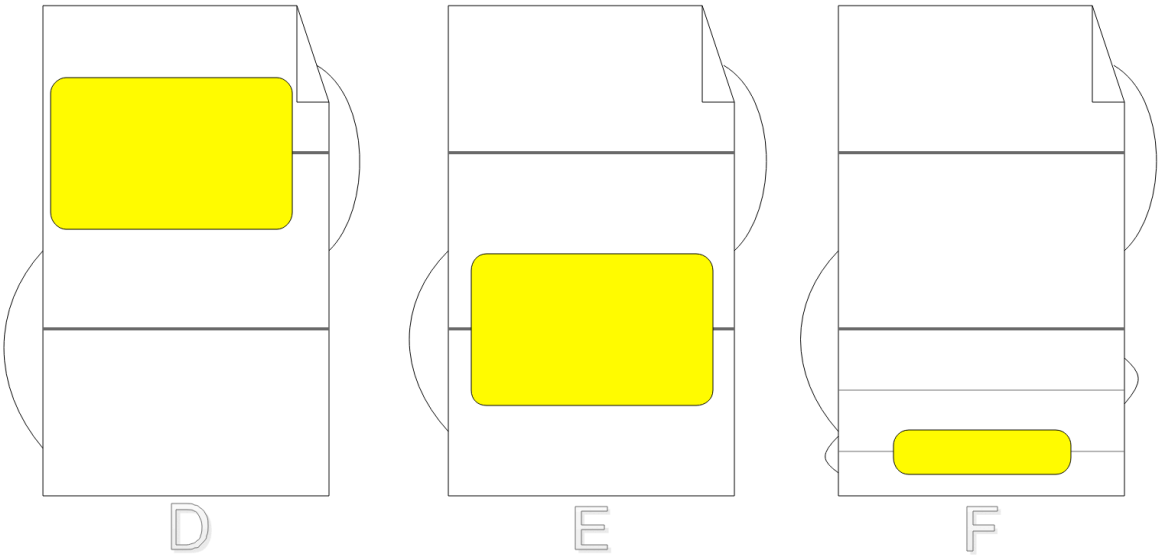

Figura 3.3: Divisão recursiva ternária do documento fonte

A seguir é apresentado o Algoritmo 3 implementado para a segmentação recursiva do documento fonte.

Algorithm 3 Divisão Recursiva Ternária do Documento

1: **PlagiarismSearch** (δ, S)
2: **Input:** $suspeito(\delta), fonte(S)$
3: **if** $|s| \leq |\delta|$ **then**
4: **if** $sim(Vocab(\delta), Vocab(S) > \Theta$ **then**
5: **return** plagiarism in S
6: **else**
7: **return** without plagiarism
8: endif
9: seja s_1, s_2, s_3 | s=$s_1 \oplus s_2 \oplus s_3 \wedge |s_1| = |s_2| = |s_3|$
10: **if** sim (Vocab (δ), Vocab($Vocab(s_1 \oplus s_2)$) > Θ **then**
11: **PlagiarismSearch** (δ, $Vocab(s_1 \oplus s_2)$)
12: **else if** sim (vocab (δ), Vocab ($s_2 \oplus s_3$)) > Θ **then**
13: **PlagiarismSearch** (δ, $Vocab(s_3 \oplus s_3)$)
14: endif
15: endif

Da linha três até a linha sete deste algoritmo, temos o teste base (condições que devem ser verificadas para que a recursividade seja aplicada). O documento fonte (representado por S) é

formado por três blocos *i.e.*, $S = s_1 \oplus s_2 \oplus s_3$. E os três blocos de S têm o mesmo tamanho *i.e.*, $|s_1| = |s_2| = |s_3|$.
Na linha dez, é aferida a similaridade entre o vocabulário δ e o vocabulário união dos blocos em que partes da zona de plágio foram "dispersadas" pela divisão dos blocos (como explicado acima). Os blocos s_1 e s_2, são o resultado da segmentação do documento fonte s. Entretanto, o vocabulário união será formado pela coleção de palavras que formam os blocos s_1 e s_2.
Tomando como exemplo a ilustração da imagem D da Figura 3.3, este vocabulário união será formado por $Vocab(D_1)$ e $Vocab(D_2)$.
A similirade é testada através de Θ que é o *threshold* definido (*e.g.,0.8*), considerando desta forma se 80% de δ está em um ou em dois blocos de s estamos diante de uma caso de plágio.
Entretanto, a função *PlagiarismSearch* é chamada recursivamente sempre que o tamanho dos blocos resultantes da anterior segmentação forem maiores que o δ e Θ maior ou igual que o *threshold*.

Todavia, como referimos acima, o vocabulário $Vocab(\delta)$ representa o vocabulário da frase suspeita e o vocabulário $Vocab(S)$ representa o vocabulário do documento fonte que está ser analisado naquele exato momento.
Conduto, consideramos sempre que o documento suspeito é um segmento (i.e., frase δ) e o documento fonte s é segmentado recursivamente nas duas abordagens. Na primeira abordagem, o documento fonte é segmentado em dois grandes blocos (*i.e.* $s = s_1 \oplus s_2$), cada bloco é transformado em um vocabulário. Pesquisa-se a similaridade dos vocabulários de cada bloco em relação a frase suspeita δ. A pesquisa da similaridade dos vocabulários é feito utilizando a Equação 3.1. Identifica-se a percentagem de δ que está contida em s. Se justificar uma nova divisão dos blocos de acordo as razões referidas acima, os dois blocos são novamente segmentados e a pesquisa é de novo realizada em cada um dos novos blocos resultantes.
Um dos problemas verificado e apontado acima é o facto de um bloco que contem o segmento plagiado ser segmentado justamente na região plagiada. Isto pode implicar a impossibilidade de detetar o segmento plagiado no seu todo ou partes do mesmo.

Na segunda abordagem da métrica recursiva, desta vez ternária, a segmentação do documento fonte é realizada em três blocos, (*i.e.* $s = s_1 \oplus s_2 \oplus s_3$). Feita a segmentação em três blocos criamos um vocabulário união dos blocos $s = s_1 \bigcup s_2$ e outro $s = s_2 \bigcup s_3$. Esta união entre os vocabulários permite evitar a possibilidade de uma frase plagiada não ser detetada na totalidade. O que resolve o problema inicialmente apontado.
Quer na abordagem da divisão binária assim como na ternária não consideraram *matches* simplesmente quando há os mesmos léxicos. As palavras que são sinónimos, antónimos também são detetadas. Nas experiências ficou provado que esta métrica com implementação recursiva é adequada tanto para pequenos segmentos de textos extraídos de um documento de grande dimensão, como em dois documentos de textos com tamanhos equiparados. Os casos de plágio com paráfrases são também aqui resolvidos.

Com a combinação do Algoritmo 3 e a métrica 3.1 detetamos *symmetrical paraphrase* e *asymmetrical paraphrase* [CDB07]. Um dos exemplos de paráfrases detetada nas experiências realizadas é a que se segue:[2]:

[2]paráfrases extraídas do *Google News*

Original:

French school students will be **banned** from using **mobile phones** anywhere on **school grounds** from **September**, after the **lower house** of **parliament passed** what it **called** a **"detox" law** for a **younger generation increasingly addicted** to **screens**.

Paráfrase:

From **September pupils** in **France** will **effectively** be **banned** from using **mobile phones** in **schools.**

As palavras realçadas são as mais informativas. As outras, são *stopWords* e foram removidas no ato de pré-processamento dos texto (etapa em que todas as letras são transformadas em minúsculas, a pontuação, os traços, os números e as *stopWords* são removidas, bem como outras práticas da PLN).

Portanto, a métrica da similaridade entre os vocabulários com *divisão recursiva* quer de versão *binária* como na *ternária* é aplicada unicamente ao documento fonte. As insuficiências identificadas com a *divisão recursiva binária* do documento fonte ficaram ultrapassadas com a *divisão recursiva ternária* do documento fonte.

3.4 Word2Vec

Com o aprimorar das técnicas de plágio [ASA12], a abordagem clássica BOW tem mostrado alguma insuficiência em detetar casos de plágio quando há ofuscação e, em alguns casos deteta similaridade em falsos positivos [MSC+13, AIAA15, BNM17, ASI+17] (como no exemplo referido na Secção 2.3). Com objetivo de inferir qualquer relação semântica de uma palavra com outra, duas frases textuais ou um par de palavras frásicas ou não frásicas através da sua representação vetorial assim como detetar palavras utilizadas no mesmo contexto em relação as outras da sua vizinha (palavras próximas si), outras tendências são seguidas e diversos algoritmos têm sido desenvolvidos para gerar *embeddings* tais como: *Global Vectors for Word Representation* (GloVe) [PSM14, LZM15], *sense2vec* [TML15] e *word2vec* [MSC+13, MYZ13, MCCD13, LM14, LZM15, KSKW15]. Nesta Secção fazemos uma breve incursão sobre a utilização do *word2vec* na DAP e como foi implementado nessa dissertação.
O *word2vec* é uma técnica desenvolvida por Miklov *et al*. [MCCD13], em que uma rede é treinada com um *corpus*, onde cada palavra do *corpus* tem o seu significado representado através de um *vetor semântico*. O *corpus* utilizado nessa dissertação tem $706\,978$ palavras na língua inglesa em que cada uma dessas palavras forma um vetor de dimensão 300 que determinam a sua direção semântica [MCCD13].
O fato de usarmos vetores para representar palavras significa que temos ao dispor várias operações vetoriais cujo resultado está relacionado com a semântica das palavras.
Neste trabalho, baseamos-nos em [KSKW15], e desenvolvemos um Algoritmo para DAP capaz de identificar alto nível de ofuscação utilizando *operações vetoriais* sem recursos aos cálculos matriciais comummente feitos, o que permitiu a redução da complexidade.

3.4.1 Adição de Vetores

Ao realizarmos adição de *vetores palavra*, o vetor resultante, chamado *vetor segmento*, representa a semântica das palavras somadas, o que nos permite tratar e usar vetores não só de

palavras individuais, mas conjuntos sem perder a informação semântica. Esta combinação é efetuada a partir de adição vetorial de cada *vetor palavra*, como no exemplo da Figura 3.4.

Figura 3.4: Adição de vetores palavras com sentido semântico

Adicionando esses vetores, o *vetor semântico* resultante será: **evening saturday** ou **saturday evening**.

3.4.2 Similaridade Semântica de Vetores

Para calcular a similaridade semântica entre dois vetores, devemos partir da definição expressa na Equação 3.3.

$$SimCos = a.b = ||\vec{a}||.||\vec{b}||cos(\Theta) \quad (3.3)$$

Para obter essa semelhança, é necessário calcular o produto interno entre os dois vetores normalizados. Esta operação devolve um escalar entre 1 e -1, em que 1 significa que os vetores são idênticos semanticamente, e -1 significa que são o menos similar possível.

Para normalizar um vetor, dividimos a magnitude (comprimento) do vetor formado por cada um dos seus componentes ou seja, os pesos (*que determinam o valores semânticos de cada palavra*), obtidos a partir da distância cartesiana do vetor à origem. Como ilustra a Equação 3.4.

$$SimCos(\Theta) = \frac{\vec{a}}{||\vec{a}||}.\frac{\vec{b}}{||\vec{b}||} \quad (3.4)$$

Na Figura 3.5, temos ilustrado quatro *vetores palavras* que, se calcularmos a similaridade entre essas palavras individualmente, teremos duas palavras muito similares e o *vetor palavra* **Lisbon**, resultará no *vetor semântico* menos similar.

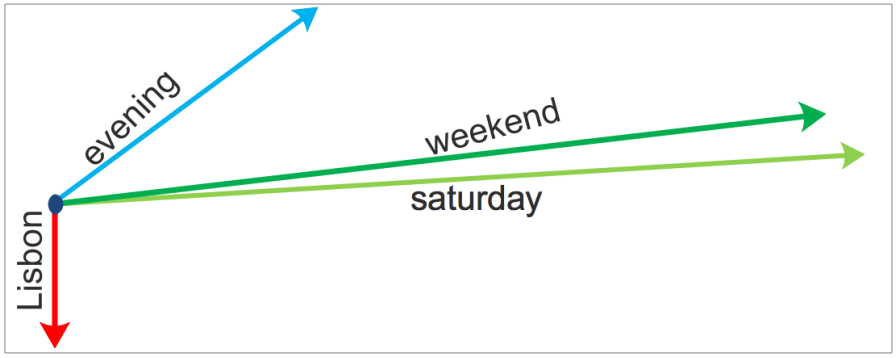

Figura 3.5: Direção semântica das palavras que formam as duas frases

Quanto mais apontarem na mesma direção, mais semelhantes são, caso contrário, menos semelhantes são. Isto é, menos similaridade existe entre os dois vetores em comparação.
De salientar que as Figuras 3.4 e 3.5 são uma ilustração, já que os reais vetores têm dimensão 300.
Tendo em conta a especificidade da nossa implementação, designámos a nossa abordagem de *Pesquisa de Plágio por Scanning Vetorial* (*PPSV*). Os detalhes da nossa implementação estão descritos na Subsecção 3.4.4.

3.4.3 Pesquisa de Plágio por Matrizes

Antes da apresentação da nossa abordagem anunciada na Subsecção 3.4.2, apresentámos a forma mais comum de utilização do *word2vec* na DAP, com recurso aos cálculo matriciais.
Normalmente o *embedding* é formado por uma matriz $\mathbf{X} \in \mathbb{R}^{d \times n}$ para um tamanho finito de vocabulário de n palavra. A i-éssima coluna, $\mathbf{x_i} \in \mathbb{R}^d$, representa o *embedding* da i-éssima palavra da coluna no documento d [KSKW15, Dan17]. Kusner *et al.* [KSKW15] apresentam a técnica de deteção de similaridade documental designada WMD. Esta técnica segue os seguintes pontos:

Normalized Bag-Of-Words (nBOW): nesta técnica, os dois documento em comparação são apresentados como vetor normalizado de Bag-Of-Words, com remoção das *stopWords*.

Word travel cost (custo de transporte de palavra): para identificar a distância semântica existente entre uma palavra do documento d e outra palavra próxima a essa em outro documento d', há um custo. Este custo associado a essa comparação, designa-se por **word travel cost**.

Document distance (distância do documento): permite que cada palavra no documento $\mathbf{d}$ seja transformada em qualquer palavra em $\mathbf{d}'$ na sua totalidade ou em parte. No espaço Euclidiano, eles definem a distância entre dois documentos como o peso mínimo cumulativo do custo necessário para "comparar" todas as palavras de $\mathbf{d}$ com $\mathbf{d}'$. Fazendo isso, a complexidade máxima no melhor caso, para o Algoritmo de WMD é $O(p^3 log\ p)$.

Relaxed Word Moving Distance (RWMD): técnica usada para otimização do Algoritmo WMD em que, tendo no documento d uma palavra i, procura-se em d' a palavra j mais similar que i. Após a otimização com o Relaxed Word Moving Distance (RWMD), a complexidade de tempo WMD baixou para $O(p^2)$.

Devido aos problemas de deteção de plágio a usar matrizes, foi desenvolvido nesta dissertação um método com melhor escalabilidade do que o método com matrizes. Os detalhes da implementação são apresentados a seguir.

3.4.4 Pesquisa de Plágio por Scanning Vetorial (PPSV)

A Pesquisa de *Plágio por Scanning Vetorial* (PPSV) é uma abordagem que utiliza *word2vec* sem recursos aos cálculos matriciais, que se fundamenta no Algoritmo 4. Desenvolvemos este método com objetivo de reduzir a complexidade dos algoritmos de deteção de plágio, aumentando o nível de eficiência e eficácia no processo de deteção de plágio, cujos detalhes relevantes de implementação são apresentados a seguir.
Dado um *segmento suspeito* com δ palavras e um *documento fonte* com S palavras, comparamos o vetor suspeito formado pela soma dos *vetores palavras* no documento suspeito, com todos os *vetores segmentos* formados por δ palavras sequencias no documento fonte um total $S - \delta$ *vetores*.
Inicialmente, na nossa implementação fizemos a busca por similaridade utilizando a similaridade cosseno (Equação 3.4). Ao fazer isto, o algoritmo não é melhor do que o das matrizes, visto que o numero de *vetores segmentos* do fonte considerados aumenta quadraticamente conforme S aumenta, e para calcular cada um desses vetores, é necessário fazer δ adições de vetores (ver Equação 3.4).
Esta solução só é viável devido à possibilidade de uma otimização que diminui a complexidade algorítmica da implementação ingénua para uma que cresce quadraticamente. Esta otimização baseia-se no facto que as combinações de palavras consideradas são sequenciais, logo, com exceção da primeira, conseguem ser calculadas a partir da anterior bastando uma soma e uma subtração em vez de δ somas.
Outra otimização possível vem do *vetor segmento suspeito* ser fixo, não mudando de uma iteração para outra, logo, o seu *vetor unitário* consegue ser pré-calculado, tal como apresentamos na Equação 3.5 (vetor unitário pré-calculado); o que remove uma multiplicação por cada *vetor segmento* do fonte, e acrescenta o custo de normalização do *vetor segmento* suspeito[3]. A nossa medida de similaridade cosseno simplifica-se na Equação 3.5, uma vez que $||\hat{\delta}|| = 1$.

$$SimCos = cos(\Theta) = \frac{\hat{\delta}.\vec{s}}{||\vec{s}||} \quad (3.5)$$

Visto que o *vetor segmento* fonte é calculado de iteração a iteração e a sua constituição não depende da sintaxe, passámos a chama-lo por **vetor semântico**.

De uma forma mais detalhada, as comparações entre o vetor δ que representa o nosso *vetor segmento* suspeito e o vetor s que representa o *vetor semântico* fonte (*vetor segmento fonte*), é realizado da seguinte maneira:

1. no primeiro momento gerámos os *vetores segmento* (suspeito e fonte, processo descrito na Subsecção 3.4.1. O *vetor frase suspeito* é constante, e já o *vetor fonte* antes da sua normalização é guardado para uso posterior);
2. calculamos o produto interno e normalizamos (processo descrito na Subsecção 3.4.2);

[3]Que envolve cálculo de n somas e n quadrados e uma raiz quadrada.

3. calculamos a similaridade entre entre o *vetor segmento* e a direção de cada um dos *vetores semânticos* do segmento produzido na primeira iteração;

4. na segunda iteração, recorremos ao *vetor segmento* guardado anteriormente e removemos a primeira palavra esquerda e adicionamos a próxima palavra a direita do documento fonte ao novo *vetor segmento* e guardamos este *vetor semântico* antes da normalização. realizámos a etapa 2, 3 respetivamente. O processo se repete até ao fim.

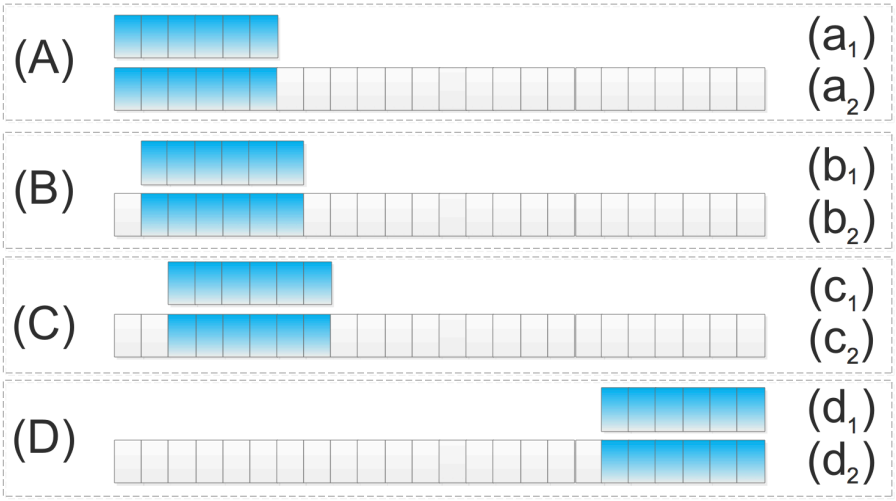

Figura 3.6: Pesquisa de Plágio por Scanning Vetorial

Na imagem (A) da Figura 3.6 temos dois *vetores semânticos* em que o vetor a_1 representa o *parágrafo suspeito* e no documento fonte (representado pela zona cinza) é extraído o *vetor semântico* a_2 (de cor azul) a partir de segmentos com o mesmo comprimento *parágrafo suspeito*, a_1. Os quadrados representam as várias palavras do documento fonte. Quando é gerado o primeiro segmento fonte (*vetor parágrafo*), é guardado uma cópia e posteriormente usado. A outra cópia é normalizada e é utilizada na primeira iteração descrita anteriormente e representada pela imagem (A).
Na imagem (B) da referida figura temos a ilustração da segunda iteração. Recorre-se ao segmento guardado remove-se a primeira palavra de *vetor semântico* guardado e adiciona-se a palavra à direita da zona azul; a próxima palavra do documento fonte e, fizemos o *scanning* como mostra a Figura 3.6 na imagem (B).
Na terceira iteração, isto é, na imagem (C) segue-se o mesmo procedimento que na iteração anterior e sucessivamente até a imagem (D).
Na imagem (D) o *vetor paragrafo* d_1, o seu *comprimento* alcança o fim do documento e faz a última comparação. Esta comparação é feita subtraindo em d_2 o comprimento de d_1 desde a extremidade direita de d_2 para se obter o último *vetor semântica*, o d_1 ilustrado na Figura 3.6 na imagem (D).
Esta comparação é feita em cada frase suspeita em relação à um documento fonte. Feito isso, percorremos todo documento fonte e desta forma, evitamos a situação de perda de informação do documento fonte se o transformássemos em um *vetor esparso*.
Todavia, o facto de não gerarmos em cada iteração um *vetor semântico* fonte a partir do zero, otimizamos. Outra otimização é obtida pelo facto de utilizarmos um *vetor unitário*, resultante do pré-calculo do *segmento suspeito*, tornam o nosso algoritmo com *complexidade de tempo*

linear, *i.e.*, $O(n)$.

Temos a seguir um exemplo de dois *vetores semânticos* afim de identificar a sua similaridade; o *vetor semântico* δ representa o segmento suspeito, e s representa o segmento fonte em cada iteração. Este último não é necessariamente um parágrafo visto que o seu módulo varia em função do comprimento do *vetor parágrafo suspeito*, *i.e.* $|\delta| = |s|$.
Seguindo o mesmo raciocino da Figura 3.6, apresentamos a seguir um exemplo com pseúdo-frases:

δ	[united	states	losing]						
	0	1	2						
S	[security	united	states	potential	spread	nuclear	weapons	destructive	war]
	0	1	2	3	4	5	6	7	8

Estas dois pseúdo-documentos[4], δ constituído por 3 palavras e o S por 9 palavras e em que cada palavra representa a sua posição no segmento suspeito e no documento fonte respetivamente.

A formação do *vetor segmento suspeito* $\vec{\delta}$ faz-se através da soma dos vetores das k palavras do segmento considerado, tal como ilustrado na Equação 3.6;

$$\vec{\delta} = \vec{\delta_0} + \vec{\delta_1} + \vec{\delta_2} + ... + \vec{\delta_k} \quad (3.6)$$

de seguida, é normalizado com base na Equação 3.7

$$\hat{\delta} = \frac{\vec{\delta}}{||\vec{\delta}||}, \quad (3.7)$$

em que $\hat{\delta}$ é o *vetor normalizado do segmento suspeito*, designado por *vetor unitário*, mantido constante em cada iteração.

Por outro lado, de forma genérica, o segmento fonte é representado segundo o expresso na Equação 3.8 ilustrada a seguir:

$$\vec{\sigma_i} = \vec{S_i} + \vec{S_{i+1}} + \vec{S_{i+2}} + ... + \vec{S_{i+k}} \quad (3.8)$$

em que $\vec{\sigma_i}$ representa a i-éssima iteração desde a palavra i até a palavra $i+k$, onde k representa o comprimento do segmento ($\vec{\delta}$)

$$simCos(\delta, S_{0:i}) = \frac{\hat{\delta}.\vec{\sigma}_0}{||\vec{\sigma}_o||} \quad (3.9)$$

Em que $S_{i:i+k}$ é o i-éssimo *vetor do segmento* que vai da palavra i à palavra k. Sendo assim, passamos a demonstração:

- Primeira iteração:

[4]Segmentos extraídos da página do facebook de Barack Obama em reação à decisão do presidente Donald Trump, em acabar com o Acordo JCPOA

Nesta iteração assim como as subsequentes, o *vetor segmento* $\vec{\delta}$ que representa o suspeito e o s é segmento extraído do documento fonte S. Como se pode ver o *vetor segmento suspeito* começa na posição 0 e vai até a posição 2.

δ	[united	states	losing]					
S	[security	united	states]	potential	spread	nuclear	weapons	destructive	war
	0	1	2	3	4	5	6	7	8

Para a primeira iteração, recorrendo a Equação 3.6; para o *vetor segmento suspeito* δ temos:

$$\vec{\delta} = \vec{\delta_0} + \vec{\delta_1} + \vec{\delta_2}$$

Após isso, normaliza-se vetor com base a Equação 3.7, dividindo cada componente do vetor δ pela sua magnitude (comprimento do vetor) para obtenção do *vetor unitário* $\hat{\delta}$. Posteriormente, por um processo similar formamos o *vetor segmento fonte* $\vec{\sigma}$ com base na Equação 3.8:

$$\vec{\sigma_0} = \vec{s_0} + \vec{s_1} + \vec{s_2}$$

Em que s_0 é a primeira palavra do documento fonte (*i.e., security*), s_1 a segunda palavra (*i.e., united*) até s_2 (*i.e., state*).

Formado o segmento fonte, utilizamos a Equação 3.10 para, primeiro normaliza-lo e posteriormente calcular o grau de semelhança utilizando a similaridade do cosseno, apresentado a seguir:

$$simCos(\delta, S_{0:2}) = \frac{\hat{\delta}.\vec{\sigma_0}}{||\vec{\sigma_o}||} \quad (3.10)$$

O $\vec{\sigma_0}$ representa a primeira palavra do segmento suspeito que é também a primeira palavra do documento fonte.

- Segunda iteração:

  Na segunda iteração o processo de formação até a normalização do *vetor unitário* $\hat{\delta}$ não é realizado, porque reutiliza-se os cálculos na iteração anterior, o que permite otimização.

  Quanto ao segmento fonte s, com os *vetores palavras* somados na iteração anterior guardados antes da normalização, remove-se a primeira palavra deste segmento e adiciona-se a próxima palavra do documento fonte ao segmento fonte S e este *vetor semântico* se desloca uma posição a direita. Assim como apresentamos a seguir na Equação 3.8:

δ		[united	states	losing]					
S	security	[united	states	potential]	spread	nuclear	weapons	destructive	war
	0	1	2	3	4	5	6	7	8

$$\vec{\sigma_1} = \vec{s_1} + \vec{s_2} + \vec{s_3}$$

Em que σ_1 é a segunda palavra do documento fonte *i.e., united*, e σ_3 *i.e., potencial* é a última palavra do *vetor semântico* analisado nesse momento.

Após a formação do segmento suspeito, é igualmente normalizado e efetua-se os cálculos da similaridade do cosseno utilizando a Equação 3.9 apresentado a imediatamente:

$$simCos(\delta, S_{1:3}) = \frac{\hat{\delta}.\vec{\sigma}_1}{||\vec{\sigma}_1||}$$

Da mesma forma que na iteração anterior, o $\vec{\sigma}_1$ representa a segunda palavra do segmento fonte.

As próximas iterações baseiam-se nos mesmos procedimentos apresentados acima, por isso, passámos à última iteração.

- Última comparação

Para este caso, quando estamos na palavra da posição do documento fonte S, porque o cumprimento do *vetor segmento suspeito* é igual a 3, chegamos ao fim do *scanning* do documento fonte S.

δ							[united	states	losing]
S	security	united	states	potential	spread	nuclear	[weapons	destructive	war]
	0	1	2	3	4	5	6	7	8

Sendo assim, utilizando as Equações 3.8 para agrupar as últimas palavras do documento fonte fazendo a soma dos *vetores palavra* a partir da posição 6 até a posição 8:

$$\vec{\sigma}_6 = \vec{s}_6 + \vec{s}_7 + \vec{s}_8$$

Feita a soma dos vetores, calculámos o *produto interno* de cada palavra que componha σ_6 e normalizamos utilizando a Equação 3.9, onde temos:

$$simCos(\delta, S_{6:8}) = \frac{\hat{\delta}.\vec{\sigma}_6}{||\vec{\sigma}_6||}$$

Ao longo deste processo as características do *vetor semântico* suspeito continuam as mesmas. As características que variam em cada iteração são as do *vetor semântico* fonte, atendendo as sucessivas subtrações e adições de palavras seguidas de outras operações matemáticas que precedem o calculo da *similaridade do cosseno*.

Na Figura 3.7, é mostrado um gráfico, relativo a quatro documentos suspeitos, designadamente s_1, s_2, s_3 *e* s_4 extraídos de um documento fonte. Os documentos s_1, s_2 s_3 foram modificados na posição 210 do documento fonte.

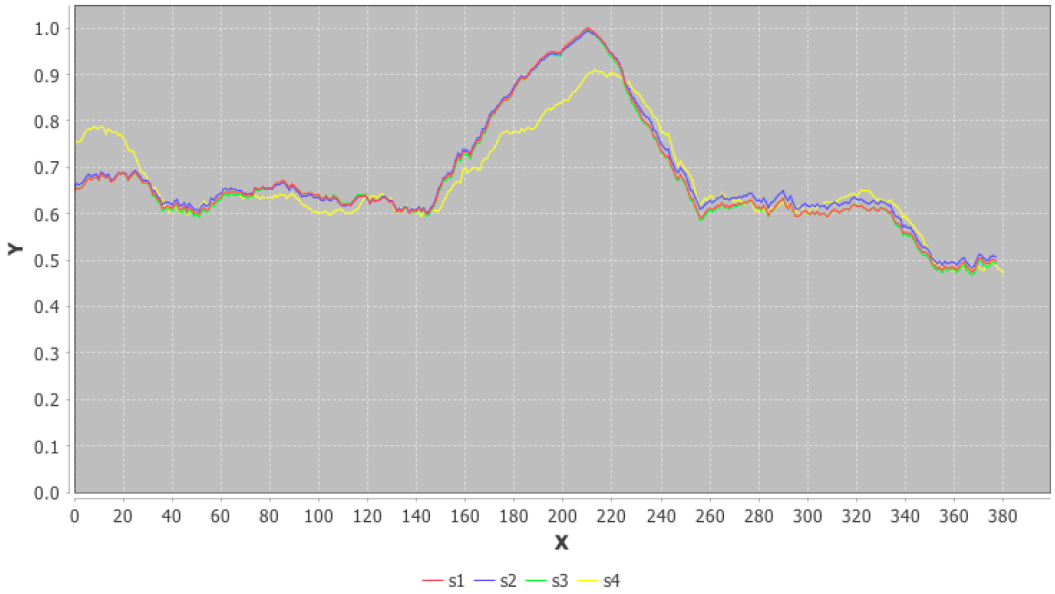

Figura 3.7: Ilustração dos Resultados Utilizando PPSV

Feitas as comparações, nesta figura temos os resultados interpretados da seguinte forma: no eixo das ordenadas temos os valores que determinam o grau de similaridade entre os documentos suspeitos em relação ao fonte e no eixo das abcissas, o comprimento (tamanho) dos documentos analisados. A partir da posição 150 do documento original (fonte) começa a crescer o valor da similaridade. No documento s_4 em que há mais modificações o valor da similaridade está em 0.9 e, onde houve modificações de uma só palavra *i.e.*, s_1, s_2, s_3 o valor a similaridade atinge o máximo, (i.e., 1.0). Entretanto, não precisa ser um *correspondência exata* para ser detetado a similaridade, como podemos ver no Gráfico do documento s_4.
A implementação do *word2vec* da nossa abordagem é baseado no Algoritmo 4 que é um dos dois métodos propostos nessa dissertação.

Algorithm 4 PPSV

Input: $word2vec, \delta, d$
$\delta = [v_1, ..., v_n]$
$d = [w_1, ..., w_k]$
$\delta \leftarrow \sum_{i=1}^{k} word2vec(v_i)$
$\hat{\delta} = \frac{\vec{\delta}}{||\vec{\delta}||}$
$\sigma_i \leftarrow \sum_{i=1}^{k} word2vec(w_i)$
for $i \in \{1, ..., n-k+1\}$ **do**
    $simCos \leftarrow \frac{\hat{\delta}.\vec{\sigma_i}}{||\vec{\sigma_i}||}$
    if $simCos >= \Theta$ **then**
        return PLÁGIO
    endif
    $\vec{\sigma_i} \leftarrow \vec{\sigma_i} - word2vec(w_{i-1} + word2vec(w_{i+k+1})$
endfor

Neste Algoritmo que representa o nosso principal método proposto, logo no início, carregamos o *corpus* de *vetores palavras* (*word2vec*) e os documentos fonte d (constituído por w palavras de w_1 até w_k) e carregamos também o segmento suspeito δ (constituído de v palavras de v_1 até v_n). O documento suspeito é particionado em segmento e analisamos cada um destes segmentos.
E, o $\vec{\sigma_i}$ é o i-éssimo *vetor segmento* do documento fonte, comparado, ilustrado na Equação 3.8.

Durante as nossas experiências, tivemos vários exemplos de deteções de pares de frases. Um deles é apresentado na Figura 3.8.

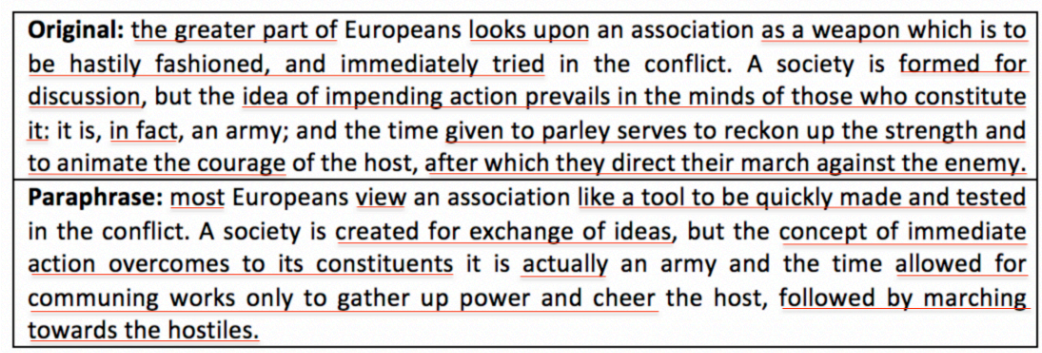

Figura 3.8: Deteção de plágio com frases parafraseadas.

As palavras destacadas nesta figura são as singulares em cada um dos segmentos e as não sublinhadas são as comuns em ambos os segmentos, embora algumas delas sejam *stopWords*.
Devido aos problemas de otimização na DAP a usar matrizes, foi desenvolvido nesta dissertação um método com melhor escalabilidade do que o método com matrizes sem necessidade de cortar os documentos fonte.

3.5 Recursos e Ferramentas Utilizadas

Nesta Secção, apresentamos os recursos e ferramentas utilizadas na realização da nossa pesquisa. De um modo geral, descrevemos as bibliotecas, a ferramenta e/ou tecnologias e sua finalidade.
Existem implementações que nos levam muito tempo e os resultados ficam longe daquilo que esperamos. Isso aconteceu connosco quando implementamos um métodos para segmentar os textos em frases. Para casos que já existem recursos experimentados, convém recorrer a eles. Pode-se dar o caso de que, algumas ferramentas existentes não nos forneçam os resultados esperados. No geral, optar por ferramentas muito utilizadas, pode significar a possibilidade de encontrar vários recursos experimentados.
Neste trabalho, nós utilizamos a Linguagem de Programação Java[5], uma das mais utilizadas e com muitos recursos e bibliotecas disponíveis. A Integrated Development Environment (IDE) escolhida foi *Netbeans*[6].

Quanto as bibliotecas, fizemos recursos à algumas para realização de tarefas específicas que a seguir descrevemos:

- *Hultig*[7] [CDB07] para as tarefas de pré-processamento de textos, *i.e.*, leitura dos documentos, lematização, remoção dos sinais de pontuação e outras boas práticas da PLN.
- *OpenNLP* [8], utilizada para segmentação dos textos em frases.

[5]https://www.java.com/en/
[6]https://netbeans.org
[7]http://www.di.ubi.pt/ jpaulo/hultiglib/
[8]$https://www.tutorialspoint.com/opennlp/opennlp_sentence_detection.htm$

- Para deteção dos sinónimos de uma palavra (referenciados no Algoritmo 2) e para a *stemming* (reduzir uma palavra ao seu radical), utilizamos a biblioteca **WordNet JWI** [9].
- Recorremos a utilização da biblioteca *JFREE-CHART* [CHA] para a construção do Gráfico 3.7.

Além das bibliotecas, a principal ferramenta utilizada para a realização das experiências da tarefa de *análise detalhada* de texto de forma automática (deteção automática de plágio em documentos de textos), foi o *Corpus* PAN12 [10]. Ainda sobre esta mesma tarefa, para uma outra abordagem, *word2vec* [MSC+13], utilizamos um corpus treinado com $706\ 977$ palavras, tendo um pouco mais de $2,03$ GB de tamanho em *plain-text*.

3.6 Sumário

No presente capítulo, realizámos experiências de *análise detalhada de documentos* utilizando três métodos (pesquisa linear, análise recursiva de documentos e PPSV, e duas técnicas (métricas da similaridade entre vocabulários e similaridade de Jaccard). A análise recursiva do documento foi feita em duas abordagem (binária e ternária). Recorremos a ternária em função de um problema identificado com a análise recursiva binária. Além disso, apresentámos uma técnica de criação de vocabulário presente no Algoritmo 2.
O método PPSV é uma nova abordagem de utilização do *word2vec* sem os cálculos matriciais. Dos métodos e técnicas experimentadas tivemos vários resultados que são apresentados no Capítulo 4. Esses resultados justificam o nosso posicionamento em relação a nossa pesquisa.

[9] http://projects.csail.mit.edu/jwi/
[10] Corpus construído para as competições da PAN para as deteções de plágio [PHG+12]

Capítulo 4

Análise Comparativa dos Resultados

Os métodos e técnicas discutidos no Capítulo 3 são aqui apresentados para ajudar-nos a analisar os resultados obtidos dos três algoritmos implementados e as duas métricas de *análise detalhada do documento*.

Analisamos neste capítulo, de forma quantitativamente os resultados obtidos das medições e experiências dos métodos e técnicas utilizados quanto a *eficiência temporal* como a *eficácia de deteção* de plágios. Realizámos também uma comparação dos resultados produzidos com os existentes na literatura. Na parte final, falámos dos recursos, ferramentas e tecnologias utilizadas para a materialização da nossa dissertação.

A seguir apresentámos detalhes relevantes de alguns pontos que foram analisados neste capítulo.

4.1 Métricas para Avaliação e Validação do Sistema

Para que se saiba a eficácia dos algoritmos de deteção de plágio, por forma a avaliar a qualidade de deteções dos sistemas de DAP são usadas um conjunto métricas algumas das quais foram originalmente desenvolvidas para medir a performance da RI. Outras ajustadas a recuperação fonte do documento e análise detalhada de documentos. Neste excerto são apresentadas dois grupos de métricas utilizadas para avaliar as experiências realizadas nesta dissertação bem como validar os métodos que propomos.

Num primeiro momento para avaliação do nosso sistema e os métodos aqui propostos, recorremos ao primeiro grupo de métricas baseada na Tabela 4.1 e constituídas pelas equações 4.1, 4.2, 4.3 e 4.4, apresentadas a seguir.

Tabela 4.1: Modelo de matriz confusão para predições de classes. Baseado em [WFHP16, FC16, Dan17]

		Predição de segmentos com e sem plágio		
		Com plágio	Sem plágio	**Total de Referências**
Método	Segmentos com plágio	TP	FN	-
	Segmentos sem plágio	FP	TN	-
Total de Predições		-	-	

A Equação 4.1, **Precision** é utilizada para medir a taxa de "acertos" corretamente detetados pelo sistema [PEBc$^+$11, FC16, Dan17].

$$\mathbf{Precision} = \frac{tp}{tp + fp} \quad (4.1)$$

Accuracy ou acurácia de dados - a Equação 4.2 é utilizada para classificar qualidade de acerto do sistema [PEBc$^+$11, FC16, Dan17].

$$\mathbf{Accuracy} = \frac{tp + tn}{tp + fp + tn + fn} \quad (4.2)$$

Para medir o grau de acertos nos *inputs* (i.e., casos suspeitos de plágio que efetivamente foram plagiados), corretamente detetados, usa-se a *Recall*, Equação 4.3 [PHG+13, FC16, Dan17].

$$\mathbf{Recall} = \frac{tp}{tp + fn} \quad (4.3)$$

As Equações 4.1, 4.2 e 4.3 fazem menção de variáveis True Positive (TP), True Negative (TN), False Positive (FP) e False Negative (FN). Estas variáveis segundo o nosso contexto, os TP são aqueles segmentos que foram efetivamente plagiados e o sistema detetou-os como plágio. TN são os segmentos que não contêm plágio e o sistema não detetou plágio algum neles. FP são os segmentos que não foram plagiados e o sistema detetou-os como sendo plágio quando, na verdade não são. E por último, FN são segmentos realmente plagiados e o sistema não os detetou como sendo plágio.

A métrica **F-Measure** combina a **Precision** e **Recall** numa única fórmula formando a média harmónica entre as duas medidas; como mostra a Equação 4.4 [PSBCR10, PHG+13, Dan17].

$$\mathbf{F - Measure} = \frac{(1 + \beta^2) * Precision * Recall}{\beta * Precision + Recall} \quad (4.4)$$

Estas quatro medidas apresentadas acima, foram utilizadas para avaliação da performance dos algoritmos de deteção de plágio apresentados nesta dissertação [FC16, Dan17].

O segundo grupo métricas utilizada foram ajustadas com base nas quatro equações supracitadas [PEBc+11, PHG+13, FC16]. Estas métricas permitiram comparar os resultados da nossa pesquisa e os produzidos por três edições da PAN [PHG+12, PHG+13, PHB+14] e os de Felipe e Cordeiro [FC16] ao nível da performance dos dos métodos de *análise detalhada*.

Estas métricas são nomeadamente as equações 4.5, 4.6, 4.7, 4.8, 4.9 e a Equação 4.10 apresentadas a seguir.

$$prec(R, S) = \frac{1}{|R|} \sum_{r \in R} \frac{|\bigcup s \in S(s \sqcap r)|}{|r|}. \quad (4.5)$$

As Equações 4.5 e 4.6 servem para garantir que um caso de plágio ainda que seja detetado mais de uma vez, é reportado uma única vez [PSBCR10, PHG+13].

$$rec(R, S) = \frac{1}{|S|} \sum_{s \in S} \frac{|\bigcup r \in R(s \sqcap r)|}{|s|}. \quad (4.6)$$

$$onde : s \sqcap r = \left\{ \begin{array}{l} s \cap r, \;\; se \;\; r \;\; detetou \;\; s \\ 0 \;\; de \;\; outra \;\; forma \end{array} \right\} \quad (4.7)$$

A Equação 4.8 é utilizada para quantificar os casos de plágios detetados que foram relatados mais deu uma vez [PHG+13].

$$gran(R, S) = \frac{1}{|S_R|} \sum_{s \in S_R} |R_s| \quad (4.8)$$

A Equação 4.9 é uma combinação das métricas 4.5, 4.6 e 4.8 para uma avaliação única das abordagens de deteção de plágio [PHG+13].

$$plagdet(R, S) = \frac{F_1}{log_2(1 + gran(S, R))} \quad (4.9)$$

O valor de F_1 é a média harmónica da métrica de 4.5 e 4.6 [PHG+13].

$$F_1(S, R) = \frac{2 \times prec(S, R) \times rec(S, R)}{prec(S, R) + rec(S, R)} \quad (4.10)$$

Fora esses dois grupos de métricas, é apresentado na Secção 4.2 os resultados das medições que permitiu-nos obter das experiências realizadas a *eficiência temporal*.

4.2 Eficiência Temporal

Antes de passarmos para a avaliação e validação da performance dos algoritmos e métricas implementados nesta dissertação, ao nível da eficácia, conforme descrito na Secção 4.1, cada um dos Algoritmos e as métricas implementadas foram feitas medições para obter a eficiência temporal. Nesta Secção, apresentámos os resultados relativos à essas medições durante a realização das nossas experiências.

Como dissemos na Secção 3.1 do Capítulo 3, trabalhamos com *corpus* PAN12 [PHG+12], com uma coleção constituída por $4\ 210$ documentos fonte de extensão *.txt* e $1\ 804$ documentos suspeitos com a mesma extensão e 201 documentos de extensão *.xml*.

De modo a obtermos os tempos de execução em cada experiência, fizemos as medições; - iniciámos as nossas experiências com um conjunto de 230 segmentos suspeitos em que 220 são segmentos plagiados de facto e 10 são segmentos sem plágio numa coleção de 20 documentos que foi sendo adicionado outros documentos gradualmente durante as medições até termos fixado em $1\ 000$ o número de documentos na coleção. Em cada incremento (aumento de documentos fonte na coleção) realizado, fazíamos a pesquisa. Os tempos obtidos são apresentados na Tabela 4.2.

Tabela 4.2: Tempo de Execução das Métricas Implementadas

Files	PLinear	MRB	JRB	MRT	JRT	PPSV
20	130,934	3,606	2,809	4,710	3,408	2,788
50	178,612	5,953	3,923	7,401	5,968	4,383
100	230,095	7,232	8,634	12,363	14,505	7,937
200	267,026	10,595	9,495	19,644	16,166	13,480
300	278,497	17,463	15,942	25,575	18,373	14,016
400	421,592	23,317	23,433	37,819	28,209	17,484
500	544,106	32,845	24,613	43,553	35,021	21,721
600	565,554	34,222	29,418	51,601	40,742	33,545
700	621,446	40,959	32,328	59,006	42,791	38,472
800	759,432	46,880	43,642	72,837	45,768	43,573
900	818,269	48,923	53,491	76,374	54,876	50,738
1000	900,623	58,965	64,672	81,214	55,890	54,207

As colunas desta Tabela foram designadas pelas abreviaturas de cada *método de análise detalhada* implementado *i.e.*, Pesquisa Linear (PLinear), Métrica Recursiva Binária (MRB), Métrica de Jaccard Recursiva Binária (JRB), Métrica de Recursiva Ternária (Métrica de Jaccard Recursiva Ternária (MRT), Métrica de Jaccard Recursiva Ternária (JRT) e PPSV.

Para que tenhamos uma visão genérica das medições feitas de modo a proporcionar uma análise comparativa entre os resultados obtidos, apresentámos graficamente através da Figura 4.1.

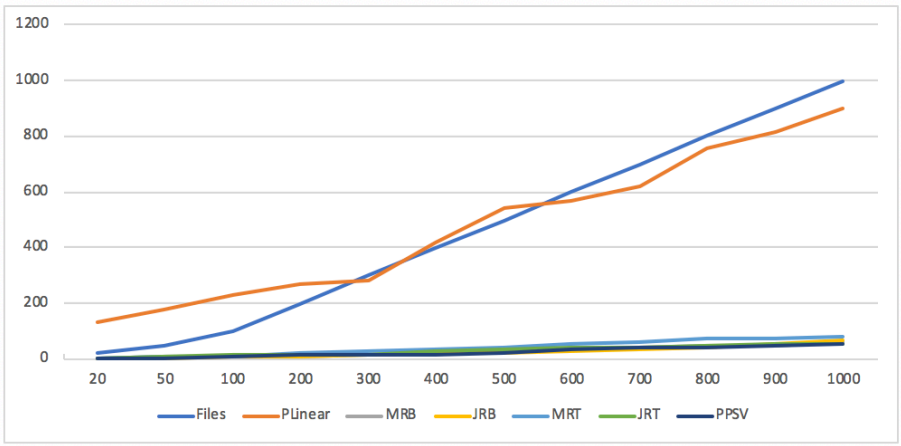

Figura 4.1: Eficiência de deteção de plágio dos algoritmos implementados

No eixo das ordenadas temos representado o tempo e no eixo das abcissas, as variações dos documentos existentes na coleção durante as doze vezes que variamos a quantidade desses para as medição da eficiência temporal, com 230 segmentos, até 1000 documentos na coleção.

As medições apresentados na Tabela acima, não representam por si só a totalidade da análise dos resultados obtidos. Outra parte complementar das experiências para quantificar e analisar os resultados das deteções de plágio é apresentada na Secção 4.3 a avaliação da precisão da eficácia.

4.3 Eficácia de Deteção

De modo a identificarmos a *eficácia de deteção* de plágio, com os dados produzidos por cada um dos algoritmos que formam o sistema desenvolvido, utilizámos nesta Secção as métricas apresentadas na Secção 4.1 para avaliação de cada parte do sistema (algoritmos e métricas implementadas).
No conjunto dos 230 segmentos suspeito temos três estratificação dos mesmos constituídos da seguinte forma: 210 segmentos com plágio exato (sem ofuscação) 10 segmentos com plágio ofuscado (modificações tipificadas como *obfuscation type high*, em português, ofuscação do tipo alto) e 10 segmentos sem plágio.
Nesta experiência, realizámos a pesquisa numa coleção com $4\ 210$ documentos fonte de extensão *.txt* e 201 documentos com anotação dos segmentos plagiados (início e fim bem como os documentos fonte e suspeitos respetivamente) de extensão *.xml*, em que o maior documento da coleção tem $177\ 134$ palavras e $13\ 742$ linhas e o seu tamanho são $994k$ já pré-processado (removidas as tabelas, as imagens, os gráficos, etc.), e o seu menor ficheiro é de $10k$ com 141 linhas e com $1\ 836$ palavras igualmente pré-processado.
Entretanto, procuramos encontrar as regiões plagiadas começando na posição onde inicia o plágio em ambos documentos (suspeito e fonte). A Figura 4.2 ilustra os resultados obtidos de forma genérica.

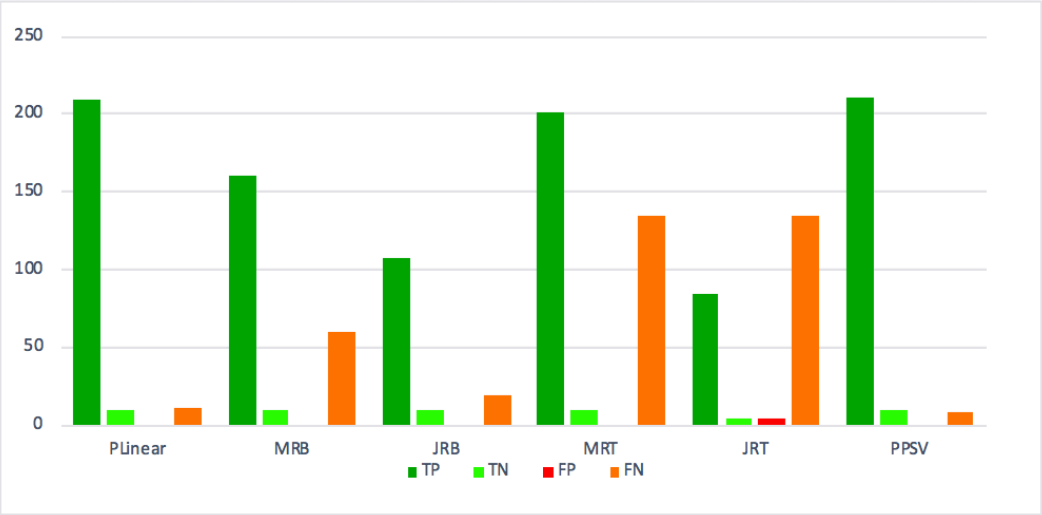

Figura 4.2: Análise comparativa dos segmentos detetados

Dividimos o sistema em partes para aferir a performance de cada um dos métodos e métricas implementadas. Nos resultados apresentados (ver Figura 4.2), o único método que deteta "falsos positivos" é a *métrica de Jaccard* na abordagem *recursiva ternária* (JRT). Outros detalhes da análise da *eficiência de deteção* são apresentados na Subsecção seguinte.

4.3.1 Análise dos Resultados Obtidos

As medições dos três algoritmos e as duas métricas implementadas bem como a suas variações, foram avaliadas as suas performances nas experiências realizadas nas duas vertentes já mencionadas (*eficiência temporal* e *eficácia na deteção*) descritas nas Secções 4.2 e 4.3. Realizámos a pesquisa na coleção com $4\ 012$ documentos fonte, com cada um dos método separadamente e os seus resultados também discutidos de forma separada e no final fazemos a comparação dos

mesmos. As etapas destes processo são apresentados em detalhes nas Tabelas 4.3, 4.4, 4.5, 4.6, 4.7, 4.8.
Os mesmos segmentos suspeito constituídos como descrito na Secção 4.3 foram utilizados em cada experiência ilustradas nas tabelas já referidas e analisadas a seguir.
Sendo assim, para o Algoritmo 1, os resultados são apresentados na Tabela 4.3.

Tabela 4.3: Matriz confusão do Algoritmo de pesquisa linear

		Segmentos com plágio	Segmentos sem plágio
PLinear	Segmentos com plágio	209	11
	Segmentos sem plágio	00	10

Os resultados obtidos da experiência são: dos 220 segmentos plagiados, 209 foram detetados como tendo sido plagiados (TP), 11 segmentos contendo plágio (FN) não foram detetados neste método. Para os 10 segmentos sem plágio foram de facto confirmados como limpos (TN) Não houve caso algum que não fosse plagiado e que terá sido detetado como sendo plagiado, ou seja nenhum (FP).
Os 11 segmentos com plágio não detetados, em 10 dos quais existem neles ofuscação e *um* não tinha a sequência mínima de palavras correspondentes determinadas, que para as experiências foram consideradas no mínimo 10 palavras. O facto deste método ser **sequencial** e detetar *matches* (correspondências) exatos, se houver reordenação de palavras, ou substituição por outras *i.e. sinónimos*, os segmentos plagiados não serão detetados.

O método analisado a seguir é a primeira abordagem da divisão recursiva binária do documento e tem os seus dados ilustrados na Tabela 4.4:

Tabela 4.4: Matriz confusão da divisão recursiva binária do documento

		Segmentos com plágio	Segmentos sem plágio
MRB	Segmentos com plágio	160	60
	Segmentos sem plágio	00	10

No conjunto de 220 segmentos plagiados, 160 foram identificados corretamente (TP), 60 segmentos com plágio não foram identificados (FN). No conjunto de 10 segmentos sem plágio nenhum foi identificado como tendo plágio (FP), isto é, os segmentos sem plágio foram confirmados como não sendo segmentos plagiados.
O que se pode dizer que dos dados produzidos pela experiência realizada com este método de **Divisão recursiva binária do documento** é: em determinadas segmentação (nas divisões recursivas) do documento fonte em blocos, houve divisões que ocorreram justamente na região plagiada e impossibilitou a deteção de plágio nestes blocos. O que também se pode dizer é que este método não detetou nenhum (TN).

Os dados apresentados na Tabela 4.5, representam a experiência realizada com a métrica de *Jaccard*, adaptada a *divisão binária recursiva do documento*.

Tabela 4.5: Similaridade de Jaccard com divisão recursiva binária do documento

		Segmentos com plágio	Segmentos sem plágio
JRB	Segmentos com plágio	107	113
	Segmentos sem plágio	00	10

Nesta Tabela para os 220 segmentos plagiados, obtivemos os resultados seguinte: 107 foram corretamente detetados (TP), nenhum segmento sem plágio erradamente identificados, ou seja nenhum (FP) e 113 segmentos com plágio foram identificados como não tendo plágio (FN). E por fim, os 10 segmentos sem plágio foram identificados como não tendo (TN).

Cerca de 48,6% foi detetado de forma errada e se combinarmos com outras variáveis produzidas nesta experiência em específico, o seu nível de acerto é muito baixo. Sendo assim, este método não é recomendável para o problema que nos propusemos a resolver.

Utilizando o Algoritmo 3, com a *Divisão recursiva ternária do documento*, os resultados são apresentados na Tabela 4.6:

Tabela 4.6: Matriz confusão do Algoritmo de divisão recursiva ternária

		Segmentos com plágio	Segmentos sem plágio
MRT	Segmentos com plágio	201	19
	Segmentos sem plágio	00	10

Com os mesmos segmentos suspeito, isto é, 220, 201 foram detetados como sendo de facto plágio (TP), e 19 não identificados corretamente (FN), nenhum (FN). Para os 10 segmentos sem plágio foram bem identificados como não tendo plágio (TN).
Pelos números produzidos, podemos afirmar que o problema identificado na utilização da abordagem de *Divisão recursiva binária do documento*, foi resolvido com a abordagem da *Divisão recursiva ternária do documento*. Contra os 48% da abordagem binária, a abordagem ternária teve a sua taxa de erro em cerca de 17,3%, embora ainda considerada alta, mostra uma redução da abordagem anterior em aproximadamente 31%.
Na Secção 5.1 perspetivamos tarefas para a melhoria desta abordagem.

Fizemos alguns ajustes no Algoritmo 3 e voltamos a utilizar com a métrica de Jaccard, desta vez com a **divisão recursiva ternária do documento**, e os resultados são apresentados na Tabela 4.6.

Tabela 4.7: Similaridade de Jaccard com divisão recursiva ternária do documento

		Segmentos com plágio	Segmentos sem plágio
JRT	Segmentos com plágio	85	135
	Segmentos sem plágio	00	10

Com 220 segmentos plagiados 85 foram detetados como tal (TP) e 135 segmentos que foram plagiados de facto e este método os marcou como não sendo (FN). Os 10 não plagiados foram corretamente identificados como não sendo plagiados (TN).
Com uma taxa de cerca de 61% de segmentos plagiados o sistema os devolveu como não sendo plágio. Pior que a abordagem de *divisão recursiva binária do documento*.

Tal como a versão *divisão recursiva binária do documento* podemos dizer que a métrica 3.2 não é adequada para este problema.
Na tabela 4.8 apresentámos os dados produzidos para a análise do último método de *análise detalhada* implementado nesta dissertação.

Tabela 4.8: Matriz confusão do Algoritmo PPSV

		Segmentos com plágio	Segmentos sem plágio
PPSV	Segmentos com plágio	211	09
	Segmentos sem plágio	00	10

Na experiência realizada, tal como nos outros métodos analisados acima, tivemos 220 segmentos plagiados onde 211 foram devidamente identificados (TP), 09 segmentos que de facto foram plagiados que não foram correta identificados (FN) e nenhum segmento que não tenha sido plagiado se foi identificado como sendo (FP). Quanto aos 10 sem segmentos sem plágio foram corretamente identificados como não sendo plagiados (TN).
Com uma taxa de acerco com cerca de 96%, este Algoritmo comparativamente aos outros teve os melhores resultados. Atendo as melhorias que se pretende fazer no futuro (ver na Secção 5.1), acreditamos que esta taxa ainda será melhorada.

Os dados produzidos durante as experiências realizadas relatados nas tabelas 4.3, 4.4, 4.5, 4.6, 4.7 e 4.8 foram sintetizados para a produção da Tabela 4.9 afim de aferir a performance dos algoritmos e as métricas implementadas nesta dissertação.

Tabela 4.9: Performance dos algoritmos e métricas implementadas para deteção de plágio

Métricas	Precision	Recall	F-Measure	Accuracy
Pesquisa Linear	**1,0000**	0,9500	0,9744	0,9522
Métrica Recursiva Binária	**1,0000**	0,7273	0,8421	0,7391
Jaccard Recursiva Binária	**1,0000**	0,4864	0,6544	0,5087
Métrica Recursiva Ternária	**1,0000**	0,9136	0,9549	0,9174
Jaccard Recursiva Ternária	0,9444	0,3864	0,5484	0,3913
PPSV	**1,0000**	**0,9591**	**0,9791**	**0,9609**

Estes dados são considerados como fundamentais para as conclusões que se impõem assim como temos vindo referir ao longo desta dissertação. Sendo assim, podemos fazer uma avaliação positiva em alguns dos métodos quanto a *eficácia detetiva* de análise detalhada aqui apresentados.

4.4 Análise Comparativa dos Resultados

Neste excerto, apresentamos os resultados existentes na literatura em comparação com os produzidos em dois dos métodos propostos resultantes da nossa pesquisa.

Tabela 4.10: Análise comparativa dos resultados das avaliações. Baseado em [PHB+14, FC16]

Métodos	plagdet	rec	prec	granularity	tempo
MRT	0,6901	0,5903	1,6803	1,0010	00:01:22
PPSV	**0,9749**	**0,9342**	**0,8977**	0,9174	**00:00:55**
[PHG+12]	0,7380	0,6780	0,8240	1,0100	00:00:00
[TR13, PHG+13]	0,8222	0,7619	0.8948	1.0014	00:01:02
[PHB+14]	0,8693	0,8578	0,8859	1,0036	00:05:31
[FC16]	0,1325	0,1832	0.5835	3.3033	01:36:00

4.5 Discussão dos Resultados

Nesta Secção descrevemos os principais pontos dos resultados obtidos e os recursos e ferramentas utilizadas nesta dissertação. As nossas reflexões têm como foco os resultados das tabelas 4.2, 4.10, 4.9, 4.10, e das Figuras 4.1 e 4.2.
Sinteticamente são aqui apresentados alguns aspetos relevantes do *estado da arte* (comparação), da implementação e sobre os resultados alcançados ao longo das experiências realizadas em torno das medições da *eficiência temporal* e da *eficácia na deteção* dos métodos de *análise detalhada de documento*. Sendo quatro algoritmos implementados (três dos quais são métodos de DAP e um mais específico, para criação de vocabulário) e duas métricas utilizadas para obter os resultados que nos posicionamos sobre os mesmos.
Para as experiências tivemos 230 segmentos suspeitos divididos da seguinte forma: 10 segmentos sem plágio, 10 segmentos com plágio ofuscado e 210 com plágio exatos (plágio sem ofuscação) obtidos de 201 documentos anotados de formato *.xml* extraídos de 158 documentos fonte.
Por outro lado, começamos as experiências em cada método e nas métricas apresentadas com uma coleção de 20 documentos. Posteriormente fomos aumentado gradualmente até atingirmos num total de $1\ 000$ para a experiência de *eficiência temporal* e $4\ 210$ documentos na coleção para a experiência de *eficiência na deteção*.

- Pesquisa Linear

  A *pesquisa linear* é um dos métodos de DAP utilizados nesta dissertação, quanto as medições, durante as nossas experiências da *eficiência temporal* foi o método maior tempo executou até encontrar o primeiro segmento plagiado. Este método já foi utilizado por outros investigadores [PHS+12]. No que concerne a eficácia na deteção de segmentos plagiados, o que constamos é que, se os segmentos plagiados são cópias exatas do segmento fonte (segmento extraído do documento fonte), este método pode ser uma boa alternativa. Mas em outras em que há plágio inteligente [ASA12, ASP15, HPS17], e admitindo que nem sempre se sabe a natureza do plágio, (se há ou não tentativas de ofuscação) pode condicionar a sua utilização.

  Nas nossas experiências, este método detetou todos os casos de plágio sem ofuscação. Mesmo removendo as *stopWords*. Visto que elas (*stopWords*), são removidas em ambos os segmentos o que, não impede a deteção de sequências. O limiar (*i.e., 10 correspondências no mínimo*) definido para este método é relativo aos dois documentos e/ou frases analisadas.

  Portanto, embora a *precision*, *recall*, *F-Measure* e na *accuracy* não tenham atingido maus valores pelo facto de termos um número reduzido de segmentos com plágios ofuscados (ver Tabela 4.9), não recomendamos para situações em que os casos de plágio têm tentativas de ofuscação, não as conseguimos detetar aí. E também por causa da eficácia temporal (ver Tabela 4.2).

- Divisão Recursiva do Documento

  A *divisão recursiva do documento*, teve algumas variações. Estas variações foram essencialmente duas abordagens distintas: a abordagem da *divisão recursiva binária* (divisão

do documento fonte em dois blocos) e *divisão recursiva ternária* (divisão do documento fonte em três blocos). Além destas, variámos também nas métricas utilizadas. Em todas as medições e experiências realizámos com a Equação 3.1 e com a Equação 3.2.

- Métrica da Similaridade entre Vocabulários

  Nesta primeira abordagem, considerámos que o documento suspeito é um segmento (resultante da divisão em partes do documento), e o documento fonte é segmentado recursivamente em dois grandes blocos e é pesquisado o plágio em cada um destes blocos em cada divisão recursiva do documento fonte. Quanto *eficácia temporal*, este método mostrou-se eficiente, porém, pelo facto da divisão do documento fonte ser ao meio, fez com que, em determinadas situações o segmento plagiado foi exatamente extraído na zona dividida. Pelo que, não detetou esses, o que reduziu a sua eficácia.

  Entretanto, se o plágio for por paráfrases, os segmentos plagiados são identificados desde que não estejam na zona da divisão como ilustrado na Figura 3.2 através da imagem C. Com os resultados obtidos, presentes na Tabela 4.9 recorremos à uma outra abordagem para mitigar este problema.

  Na segunda abordagem, *divisão recursiva ternária* do documento fonte, atendendo a intercessão entre vocabulários adotado nesta abordagem ilustrado na Figura 3.3, as situações de plágio que não eram identificadas na primeira abordagem passaram a ser localizadas como mostram as Tabelas 4.6, 4.9. Esta união entre os vocabulários nos permite evitar a possibilidade de uma frase plagiada por paráfrase não ser detetada na totalidade ainda que esteja na região em que os blocos foram segmentados.

  Inicialmente não tínhamos a criação de vocabulário. O facto de termos gerado vocabulários através da utilização do Algoritmo 2, aumentou a eficácia dos resultados, porque a tentativas de ofuscação ou a substituição de palavras pelos seus sinónimos passaram a ser detetados embora a eficiência temporal tenha diminuído.

  A eficiência temporal na primeira abordagem mostrou-se melhor que na segunda (ver Tabela 4.2 e 4.11), e a eficácia na deteção melhorou na segunda abordagem (ver Tabela 4.9). O limiar (*e.i., 0.7*) foi o definido para ambas as abordagens ao longo das suas segmentações em bloco utilizando a métrica 3.1.

  Portanto, quer na abordagem da divisão binária assim como na ternária não consideraram *matches* simplesmente quando há os mesmos léxicos. As palavras que são sinónimos, também são detetadas. Nas experiências ficaram provadas que esta métrica com implementação recursiva é adequada deteção de plágio sem ofuscação ou parafraseados em *symmetrical paraphrase* e *asymmetrical paraphrase* [CDB07] com pequenos segmentos de textos extraídos de um documento enorme ou com tamanhos equiparados. A segunda abordagem ainda pode ser melhorada (ver Secção 5.1).

- Similaridade de Jaccard

  Dentro da Divisão Recursiva do Documento, fizemos algumas adaptações e utilizamos a métrica 3.2 conhecida como *similaridade de Jaccard* [NSNW13]. Esta métrica foi utilizada nas duas abordagens descritas, com a divisão sucessivamente (binária e ternária) do documento em blocos. Os resultados mostraram, entretanto, que a Equação 3.2 é praticamente inviável em situações que o plágio é um pequeno ou grande segmento extraído de

um documento enorme, analisado recursivamente. Mesmo com reconfigurações do *limiar* (*e.i., 0.2*), detetamos um ou outro caso, (ver Tabela 4.9).

Isso se explica pelo facto desta métrica conservar no denominar apenas as palavras (assim como os sinónimos) existentes no segmento suspeito e que também estão no documento fonte, não ignora esses n-grams. O que não acontece com a métrica 3.2 porque o tamanho da intercessão é sempre menor que o tamanho da união o que torna remota a possibilidade de detetar pequenos segmentos em um documento de grande dimensão. E os dados provam isso, embora dos 10 segmentos ofuscados a métrica 3.1 detetou simplesmente um segmento (ver Tabela 4.9).

- PPSV

Partimos do pressuposto que documentos semelhantes têm *vetores semânticos* semelhantes ou similares, porque são formados por palavras similares, e consideramos que um documento é similar ao outro se um vetor ou mais vetores destes documentos são formados por palavras "muito" similares [TP10, Dan17], cada segmento suspeito é transformado em um **vetor semântico**; desse segmento que, se determina o contexto da informação a pesquisar em cada comparação com outro *vetor semântico* (similar ou dissimilar).

Diferente do tradicional *word2vec* (utilizando matrizes) [KSKW15], adotamos uma abordagem de *PPSV ao documento fonte* para otimização computacional e os resultados mostram que, para os segmentos suspeitos utilizados houve um número de deteção considerável tanto para os casos sem ofuscação (foram todos detetados) como para os ofuscados, dos 10 segmentos, 09 foram detetados e apenas 01 não foi identificado o que nesse particular mostra uma taxa de acerto de 90% em casos com ofuscação. Quanto aos segmentos sem plágio, todos foram confirmados como sendo sem plágio. De um modo geral a eficácia detetiva deste modo foi considerada positiva (ver Tabela 4.8) e a eficiência temporal também mostrou-se entre as melhores na comparação entre os métodos implementados (ver Tabela 4.2 e 4.11) como entre os existentes na literatura (ver Tabela 4.10), o que de modo geral podemos considerar como positiva. O limiar definido para este método, foi *0.92*. Pretendemos ainda numa próxima pesquisa melhorar alguns aspetos mencionados na Secção 5.1 para torna-lo mais eficaz.

Em síntese, nesta dissertação, cada um dos algoritmos e métricas implementadas tiviram os seus resultados. Uns tiveram resultados que segundo a abordagem que seguimos podemos considerar de "não satisfatório" e outros que são satisfatórios e que podem ser melhorados ainda mais em pesquisas futuras. A Tabela 4.11 mostra um panorama geral de cada um dos métodos abordados nesta dissertação.

Tabela 4.11: Eficácia e tempo médio até deteção do primeiro segmento plagiado

Métricas	F-Measure	Tempo
Pesquisa Linear	0,9744	00:08:00
Métrica Recursiva Binária	0,8421	00:00:27
Jaccard Recursivo Binária	0,6544	00:00:26
Métrica Recursiva Ternária	0,9549	00:00:41
Jaccard recursivo Ternária	0,5484	00:00:30
PPSV	0,9791	00:00:25

Em termos de eficácia o melhor resultado pertence ao Algoritmo 4 (PPSV) e em eficiência a melhor média também pertence a este algoritmo, imediatamente seguido pela métrica *recursiva binária Jaccard*.

4.6 Sumário

Os métodos e técnicas cujo relato da implementação presente no Capítulo 3, neste capítulo foram feitas as medições e experiências em cada um deles para aferir a *eficácia temporal* e a *eficiência na deteção* dos mesmos. Os resultados produzidos foram apresentados e analisados para a avaliação e validação dos algoritmos propostos através das métricas definidas pela PAN11, PAN13 e PAN14. Estas métricas possibilitaram a comparação dos dados produzidos com alguns resultados existentes na literatura sem perder de vista o objetivo definido no princípio da pesquisa, através do problema identificado. Fizemos algumas variações da métrica clássico de Jaccard [NSNW13] para eventual aperfeiçoamento e a medida que variávamos os resultados ficavam longe do esperado. Isso nos levou a deduzir que as adaptações que fazíamos não eram as mais convenientes. Já com a métrica dos vocabulários [LMD01] as variações que fomos fazendo, foram tornando o método mais eficaz e com menos eficiência a medida que a eficácia aumentava.
O que podemos considerar que as abordagens propostas resolvem efetivamente o problema mas a métrica de Jaccard não se adequa a este problema. A comparação feita com os resultados obtidos em outras pesquisas existentes na literatura justificam a nossa afirmação.
Portanto, na análise comparativa cingiu-se especificamente na *eficácia temporal* e a *eficiência na deteção*.

Capítulo 5

Conclusões e Trabalhos Futuros

Conclusão

Embora o plágio seja um problema antigo, nas últimas décadas tem vindo a modernizar as formas de o realizar atendendo à quantidade de documentos produzidos e difundidos na *Web*. Especialistas de várias áreas têm estado empenhados criando ferramentas normativas e tecnológicas para o combate a esta prática.

Além do problema da *propriedade intelectual*, estão também outros problemas tais como a *copyright, análise forense e/ou Computer Forensic Science*[1] etc. Pensando nas consequências deste mal, nesta dissertação pesquisamos abordagens eficazes e eficientes em identificação de pequenos segmentos extraídos de um documento de grande dimensão e consequente, este segmento é "manipulado" e transformado substituindo termos por sinónimos, antónimos com reduzida possibilidade da sua deteção.

Criámos um sistema de DAP e realizámos várias experiências com três métodos de *análise detalhada de documentos* nomeadamente, *pesquisa linear, divisão recursiva do documento* e o método de *pesquisa de plágio por scanning vetorial*. Este último não faz recurso a cálculos matriciais complexas, como noutras abordagens, tornando-o ótimo (ver Tabela4.2).

Quanto ao Algoritmo 3, do método da *Divisão Recursiva do Documento*, teve duas variações: na primeira variação com a *divisão binária recursiva do Documento* utilizamos a métrica de similaridade de Jaccard (Equação 3.2) e a métrica de similar dos vocabulários (Equação 3.1). Na segunda variação, com a *Divisão Ternária do Documento*, voltamos a utilizar as duas métricas supracitadas.

Na primeira abordagem, com a Equação 3.1, o documento fonte é dividido em dois blocos e a segmentação foi feita separando zonas plagiadas em blocos distintos, (*e.g.* Tabela 4.4).

Na segunda abordagem (*divisão ternária do documento*) o documento fonte é divido em três blocos e posteriormente é pesquisado a similaridade de cada vocabulário resultante da segmentação do documento fonte em relação ao vocabulário suspeito. Se tiver o mínimo de similaridade, é feito a união com outro vocabulário do bloco mais próximo (ver Figura 3.3). Se detetar similaridade é retornado o resultado; caso contrário segmenta-se novamente cada um dos blocos fontes em três sub-blocos e faz-se novamente a pesquisa (Como descrevemos na Secção 3.3).

Os dados produzidos e apresentados nas tabelas 4.6 e 4.10 mostram que esta abordagem é eficaz. Além disso, propomos um algoritmo para criação de vocabulário (ver Algoritmo 2).

Portanto, após a análise dos resultados, propomos duas abordagens que apresentaram melhores resultados durante as nossas experiências. O método de *pesquisa de plágio por scanning vetorial* utilizando *Word2Vec* (Algoritmo 4), é adequado para identificar plágio com alto nível de ofuscação. A MRT, por sua vez, mostrou ser adequado para identificação de casos de plágio parafraseado. Nessa abordagem este método melhorou a eficácia e perdeu ligeiramente a eficiência em comparação com a MRB.

[1] computação forense ou ciência forense computacional: ramo da ciência forense digital pertencente às evidências encontradas em computadores e em dispositivos de armazenamento digital.[FBI00]

5.1 Trabalho Futuro

Durante a realização desta pesquisa, encontrámos dificuldades que despertaram a nossa atenção para outros desafios no futuro. Entretanto, tencionamos continuar a trabalhar sobre DAP, desta vez com um projeto que envolve as três etapas de deteção de plágio (*Recuperação fonte, Análise detalhada do Documento e pós-processamento*) pretendemos melhorar o Algoritmo 2 de forma que além de criar um vocabulário de sinónimos possa incluir os hipónimos, hiperónimos, antónimos e outros elementos que se consideram fundamentais para deteção de situações de plágio com ofuscação para que, na sua combinação com o Algoritmo 3 na sua versão ternária possa detetar casos de plágio com alto nível de ofuscação. Na recuperação fonte, desenvolver um algoritmo robusto que com base nos vocábulos mais longos do texto e as outras palavras mais próximas a elas, se formam as chaves de pesquisa (palavras mais relevantes) a serem submetidas ao motor de pesquisa.
Por outro lado, além de trabalhar unicamente com um *corpus word2vec* [MSC+13], pretendemos trabalhar também com outros recursos, o *corpus Sense2vec* [TML15], e até treinarmos o *word2vec* em corpora especifico de plágio, para verificar se existem melhorias efetivas, em vez do modelo usado, que foi treinado com texto de noticias. Posteriormente, usar *aprendizagem automática*, com corpora de plágio, para descobrir os melhores limiares a serem utilizados.

Bibliografia

[AIAA15] Asad Abdi, Norisma Idris, Rasim M Alguliyev, and Ramiz M Aliguliyev. Pdlk: Plagiarism detection using linguistic knowledge. *Expert Systems with Applications*, 42(22):8936–8946, 2015. vi, vii, 1, 5, 6, 10, 13, 14, 17, 19, 26

[AS16] Mayank Agrawal and Dilip Kumar Sharma. A state of art on source code plagiarism detection. In *Next Generation Computing Technologies (NGCT), 2016 2nd International Conference on*, pages 236–241. IEEE, 2016. 5

[ASA12] S. M. Alzahrani, N. Salim, and A. Abraham. Understanding plagiarism linguistic patterns, textual features, and detection methods. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(2):133–149, March 2012. vi, vii, 1, 5, 6, 8, 10, 12, 13, 14, 16, 26, 45

[ASI+17] Asad Abdi, Siti Mariyam Shamsuddin, Norisma Idris, Rasim M Alguliyev, and Ramiz M Aliguliyev. A linguistic treatment for automatic external plagiarism detection. *Knowledge-Based Systems*, 135:135–146, 2017. vi, vii, 1, 5, 6, 10, 12, 13, 14, 15, 17, 19, 20, 26

[ASP15] Salha M Alzahrani, Naomie Salim, and Vasile Palade. Uncovering highly obfuscated plagiarism cases using fuzzy semantic-based similarity model. *Journal of King Saud University-Computer and Information Sciences*, 27(3):248–268, 2015. 5, 10, 45

[BCRB09] Alberto Barrón-Cedeño, Paolo Rosso, and José-Miguel Benedí. Reducing the plagiarism detection search space on the basis of the kullback-leibler distance. In *International conference on intelligent text processing and computational linguistics*, pages 523–534. Springer, 2009. 7, 19, 23

[BMB+09] Chiara Basile, Dip Matematica, Dario Benedetto, Emanuele Caglioti, Giampaolo Cristadoro, and Mirko Degli Esposti. Caglioti e.: A plagiarism detection procedure in three steps: selection, matches and 'squares. In *SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse (PAN 09*, pages 1–9, 2009. 6, 7

[BNM17] Kensuke Baba, Tetsuya Nakatoh, and Toshiro Minami. Plagiarism detection using document similarity based on distributed representation. *Procedia Comput. Sci.*, 111(C):382–387, September 2017. Available from: `https://doi.org/10.1016/j.procs.2017.06.038`. 26

[BSMJ+18] Kamil Bennani-Smires, Claudiu Musat, Martin Jaggi, Andreea Hossmann, and Michael Baeriswyl. Embedrank: Unsupervised keyphrase extraction using sentence embeddings. *arXiv preprint arXiv:1801.04470*, 2018. 11, 19

[CDB07] Joao Cordeiro, Gael Dias, and Pavel Brazdil. A metric for paraphrase detection. In *Computing in the Global Information Technology, 2007. ICCGI 2007. International Multi-Conference on*, pages 7–7. IEEE, 2007. Accessed: 2018-11-07. 12, 15, 22, 25, 35, 46

[CHA] JFREE CHART. Jfreechart. `http://www.jfree.org/jfreechart/`. Accessed: 2018-08-08. 36

[CR08] Alberto Barrón Cedenõ and Paolo Rosso. Towards the exploitation of statistical language models for plagiarism detection with reference. 2008. 7

[Dan17] Jurafsky Dan. *Speech & language processing*. Third edition draft edition, 2017. xiii, 16, 19, 28, 37, 38, 47

[EBR10] Samhaa R. El-Beltagy and Ahmed Rafea. Kp-miner: Participation in semeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, SemEval '10, pages 190–193, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. Available from: http://dl.acm.org/citation.cfm?id=1859664.1859705. 11, 15

[ES16] Nava Ehsan and Azadeh Shakery. Candidate document retrieval for cross-lingual plagiarism detection using two-level proximity information. *Information Processing & Management*, 52(6):1004–1017, 2016. 16

[FBI00] The Federal Bureau Of Investigation FBI. Forensic science communications. https://archives.fbi.gov/archives/about-us/lab/forensic-science-communications/fsc/oct2000/computer.htm, 2000. Accessed: 2018-10-08. 49

[FC15] Bruno Felipe and João Cordeiro. Detenção de plágio em dois atos. https://www.dcc.fc.up.pt/~ricroc/homepage/publications/2015-INFORUM.pdf, 2015. Accessed: 2018-03-09. 6, 10, 12

[FC16] Bruno Garcia Prata Graciano Felipe and João Paulo da Costa Cordeiro. Métodos eficientes de deteção de plágio em grandes corpora. UBI, 2016. xiii, 5, 10, 14, 37, 38, 44

[FJ18] Corina Florescu and Wei Jin. Learning feature representations for keyphrase extraction. *arXiv preprint arXiv:1801.01768*, 2018. 11

[FSGRB16] Marc Franco-Salvador, Parth Gupta, Paolo Rosso, and Rafael E Banchs. Cross-language plagiarism detection over continuous-space-and knowledge graph-based representations of language. *Knowledge-Based Systems*, 111:87–99, 2016. 5

[GGP09] Cristian Grozea, Christian Gehl, and Marius Popescu. Encoplot: Pairwise sequence matching in linear time applied to plagiarism detection ▫. *SIGIR Forum*, add(2):10–18, 9 2009. Available from: https://www.researchgate.net/profile/Benno_Stein/publication/242293556_Overview_of_the_1st_International_Competition_on_Plagiarism_Detection/links/0deec5320e6c06d558000000.pdf#page=52. 6, 7

[Gir13] Renata Giraldi. Ministra da educação da alemanha renuncia após acusação de plágio. https://noticias.uol.com.br/internacional/ultimas-noticias/2013/02/09/ministra-da-educacao-da-alemanha-renuncia-apos-acusacao-de-plagio.htm, 2013. Accessed: 2018-03-07. 1

[HEB13] Osama Haggag and Samhaa El-Beltagy. Plagiarism candidate retrieval using selective query formulation and discriminative query scoring. In *Working Notes for the CLEF 2013 Conference*, 2013. 5, 6, 10, 19, 21

[HGK+16] Gao Huang, Chuan Guo, Matt J Kusner, Yu Sun, Fei Sha, and Kilian Q Weinberger. Supervised word mover s distance. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4862–4870. Curran Associates, Inc., 2016. Available from: http://papers.nips.cc/paper/6139-supervised-word-movers-distance.pdf. 16

[HPA+17] Matthias Hagen, Martin Potthast, Payam Adineh, Ehsan Fatehifar, and Benno Stein. Source retrieval for web-scale text reuse detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 2091–2094. ACM, 2017. 6, 10, 11, 16, 19

[HPS15] Matthias Hagen, Martin Potthast, and Benno Stein. Source retrieval for plagiarism detection from large web corpora: Recent approaches. 2015. xi, 6, 8, 11

[HPS17] Matthias Hagen, Martin Potthast, and Benno Stein. Overview of the author obfuscation task at pan 2017: safety evaluation revisited. *Working Notes Papers of the CLEF*, pages 33–64, 2017. 9, 45

[HS11] Matthias Hagen and Benno Stein. Candidate document retrieval for web-scale text reuse detection. In *International Symposium on String Processing and Information Retrieval*, pages 356–367. Springer, 2011. 10, 11, 12

[Kro11] Marcelo Krokoscz. Abordagem do plágio nas três melhores universidades de cada um dos cinco continentes e do brasil. 16:745–818, 12 2011. 1

[KSKW15] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International Conference on Machine Learning*, pages 957–966, 2015. 9, 10, 16, 26, 28, 47

[LHS+12] Kong Leilei, Qi Haoliang, Wang Shuai, Du Cuixia, Wang Suhong, and Han Yong. Approaches for candidate document retrieval and detailed comparison of plagiarism detection. *Notebook for PAN at CLEF 2012*, 2012. 5, 11, 12, 14

[LM14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014. 8, 13, 26

[LMD01] Caroline Lyon, James A. Malcolm, and Bob Dickerson. Detecting short passages of similar text in large document collections. In *EMNLP*, 2001. 19, 22, 48

[LR11] Abbas Al Lawati and Staff Reporter. gulfnews culture. http://gulfnews.com/news/uae/culture/plagiarism-costs-ba-li-zayed-book-award-1.702316, 2011. Accessed: 2018-03-07. 1

[LZM15] Shaohua Li, Jun Zhu, and Chunyan Miao. A generative word embedding model and its low rank positive semidefinite solution. *arXiv preprint arXiv:1508.03826*, 2015. 10, 26

[Mal16] Rhulani Maluleka. Derivative approach for plagiarism source retrieval. In *CLEF*, 2016. 11

[Man93] Udi Manber. Finding similar files in a large file system. In *USENIX WINTER 1994 TECHNICAL CONFERENCE*, pages 1–10. http://citeseerx.ist.psu.edu/viewdoc/

download;jsessionid=163612C97E9ABA2DAA9C36C9398CF3D0?doi=10.1.1.12.3222&rep=rep1&type=pdf, 1993. 5

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 16, 19, 26

[Mey87] Charles F Meyer. Language, context and text: Aspects of language in a social-semiotic perspective: Mak halliday and ruqaiya hasan. *TESOL Quarterly*, 21(2):353–359, 1987. 16, 19

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Neural and Information Processing System (NIPS)*, 2013. Available from: https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf. 10, 13, 16, 19, 26, 36, 50

[MSY14] Peyman Mahdavi, Zahra Siadati, and Farzin Yaghmaee. Automatic external persian plagiarism detection using vector space model. In *Computer and Knowledge Engineering (ICCKE), 2014 4th International eConference on*, pages 697–702. IEEE, 2014. 16

[MYZ13] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013. 10, 16, 19, 26

[MZKG09] Markus Muhr, Mario Zechner, Roman Kern, and Michael Granitzer. M.: External and intrinsic plagiarism detection using vector space models. In *In: Proceedings of the SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse (PAN 09*, pages 1–9, 2009. 5, 6, 7

[NB17] Nurhayati and Busman. Development of document plagiarism detection software using levensthein distance algorithm on android smartphone. *2017 5th International Conference on Cyber and IT Service Management (CITSM)*, pages 1–6, 2017. 1, 15

[NLM15] Ramesh R Naik, Maheshkumar B Landge, and C Namrata Mahender. A review on plagiarism detection tools. *International Journal of Computer Applications*, 125(11), 2015. 5, 6

[NSC17] Rao Muhammad Adeel Nawab, Mark Stevenson, and Paul Clough. An ir-based approach utilizing query expansion for plagiarism detection in medline. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 14(4):796–804, July 2017. Available from: https://doi.org/10.1109/TCBB.2016.2542803. 5, 11, 12, 13, 15

[NSNW13] Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. Using of jaccard coefficient for keywords similarity. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, 2013. 15, 19, 46, 48

[Onl15] Diário Online. Relembre casos famosos de plágios acadêmicos. http://www.diarioonline.com.br/noticias/mundo/noticia-349182-relembre-casos-famosos-de-plagios-academicos.html, 2015. Accessed: 2018-03-07. 1

[PBcE+10] Martin Potthast, Alberto Barrón-cedeño, Andreas Eiselt, Benno Stein, and Paolo Rosso. Overview of the 2nd international competition on plagiarism detection. In *In Proceedings of the SEPLN'10 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse*, 2010. 5, 6, 7, 8

[PEBc+11] Martin Potthast, Andreas Eiselt, Alberto Barrón-cedeño, Benno Stein, and Paolo Rosso. Overview of the 3rd international competition on plagiarism detection. In *In Working Notes Papers of the CLEF 2011 Evaluation*, 2011. 5, 6, 7, 8, 19, 37, 38

[PHB+14] Martin Potthast, Matthias Hagen, Anna Beyer, Matthias Busse, Martin Tippmann, Paolo Rosso, and Benno Stein. Overview of the 6th international competition on plagiarism detection. In *CEUR Workshop Proceedings*, volume 1180, pages 845–876. CEUR Workshop Proceedings, 2014. xiii, 5, 6, 8, 10, 11, 14, 15, 21, 38, 44

[PHG+12] Martin Potthast, Matthias Hagen, Tim Gollub, Martin Tippmann, Johannes Kiesel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Overview of the 4th international competition on plagiarism detection. In *CLEF (Online Working Notes/Labs/Workshop)*, 2012. 5, 6, 8, 10, 11, 12, 14, 15, 19, 36, 38, 39, 44

[PHG+13] Martin Potthast, Matthias Hagen, Tim Gollub, Martin Tippmann, Johannes Kiesel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. Overview of the 5th international competition on plagiarism detection. 2013. 5, 8, 10, 11, 14, 15, 38, 39, 44

[PHS+12] Martin Potthast, Matthias Hagen, Benno Stein, Jan Graßegger, Maximilian Michel, Martin Tippmann, and Clement Welsch. Chatnoir: a search engine for the clueweb09 corpus. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1004–1004. ACM, 2012. 45

[PSBCR10] Martin Potthast, Benno Stein, Alberto Barrón-Cedeño, and Paolo Rosso. An evaluation framework for plagiarism detection. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 997–1005, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. Available from: http://dl.acm.org/citation.cfm?id=1944566.1944681. 5, 6, 7, 38

[PSE+09] Martin Potthast, Benno Stein, Andreas Eiselt, Alberto Barron-Cedeno, and Paolo Rosso. Overview of the 1st international competition on plagiarism detection. 01 2009. 6, 7, 8

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 26

[RRP+15] Francisco Rangel, Paolo Rosso, Martin Potthast, Benno Stein, and Walter Daelemans. Overview of the 3rd author profiling task at pan 2015. In *CLEF*. sn, 2015. 19

[RRP+16] Paolo Rosso, Francisco Rangel, Martin Potthast, Efstathios Stamatatos, Michael Tschuggnall, and Benno Stein. Overview of pan'16 new challenges for authorship analysis: Cross-genre profiling, clustering, diarization, and obfuscation. In *International*

Conference of the Cross-Language Evaluation Forum for European Languages, pages 332–350. Springer, 2016. 9

[SB14] Š. Suchomel and M. Brandejs. Approaches for candidate document retrieval. In *2014 5th International Conference on Information and Communication Systems (ICICS)*, pages 1–6, April 2014. 6, 11, 12

[SKS07] Benno Stein, Moshe Koppel, and Efstathios Stamatatos. Plagiarism analysis, authorship identification, and near-duplicate detection pan'07. *SIGIR Forum*, 41(2):68–71, December 2007. Available from: http://doi.acm.org/10.1145/1328964.1328976. 5, 7, 8

[SM09] Leanne Seaward and Stan Matwin. Intrinsic plagiarism detection using complexity analysis. 502:56–61, 01 2009. 5

[SPP+15] Efstathios Stamatatos, Martin Potthast, Francisco M. Rangel Pardo, Paolo Rosso, and Benno Stein. Overview of the pan/clef 2015 evaluation lab. In *CLEF*, 2015. 8, 9, 19

[SPSG14] Miguel A Sanchez-Perez, Grigori Sidorov, and Alexander F Gelbukh. A winning approach to text alignment for text reuse detection at pan 2014. 2014. 14, 15, 19, 22

[Sta09] Efstathios Stamatatos. Intrinsic plagiarism detection using character n-gram profiles. In *In: 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse*, pages 38–46, 2009. 5, 7, 15

[STV+16] Efstathios Stamatatos, Michael Tschuggnall, Ben Verhoeven, Walter Daelemans, Günther Specht, Benno Stein, and Martin Potthast. Clustering by authorship within and across documents. In *CLEF (Working Notes)*, pages 691–715, 2016. 9

[TML15] Andrew Trask, Phil Michalak, and John Liu. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. *arXiv preprint arXiv:1511.06388*, 2015. 26, 50

[TP10] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010. 19, 47

[TR13] Diego A Rodríguez Torrejón and José Manuel Martín Ramos. Text alignment module in coremo 2.1 plagiarism detector. *Notebook for PAN at CLEF*, 2013. 44

[TSV+17] Michael Tschuggnall, Efstathios Stamatatos, Ben Verhoeven, Walter Daelemans, Günther Specht, Benno Stein, and Martin Potthast. Overview of the author identification task at pan 2017: Style breach detection and author clustering. *Working Notes Papers of the CLEF*, 2017. 9

[tvi17] tvi. Autarca de torres vedras suspeito de plagiar tese. http://www.tvi24.iol.pt/sociedade/plagio/autarca-de-torres-vedras-suspeito-de-plagiar-tese, 2017. Accessed: 2018-03-16. 1

[uol12] uol. Presidente da hungria renuncia após disputa por plágio. http://www1.folha.uol.com.br/mundo/2012/04/1070499-presidente-da-hungria-renuncia-apos-disputa-por-plagio.shtml, 2012. Accessed: 2018-03-07. 1

[WFHP16] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016. xiii, 37

[ZC16] Hamed Zamani and W. Bruce Croft. Estimating embedding vectors for queries. In *Proceedings of the 2016 ACM International Conference on the Theory of Information Retrieval*, ICTIR '16, pages 123–132, New York, NY, USA, 2016. ACM. Available from: http://doi.acm.org/10.1145/2970398.2970403. 11