Disciplinarum Scientia. Série: Naturais e Tecnológicas, Santa Maria, v. 17, n. 2, p. 293-310, 2016.
Recebido em: 03.08.2016. Aprovado em: 26.09.2016.

ISSN 2176-462X

UTILIZAÇÃO DE MINERAÇÃO DE TEXTO NA DETECÇÃO DE PLÁGIO EM TRABALHOS ACADÊMICOS[1]

THE USE OF TEXT DATA MINING TO DETECT PLAGIARISM IN ACADEMIC ASSIGNMENTS

Caroline Aquino Dias[2] e Sylvio André Garcia Vieira[3]

RESUMO

Com o aumento da utilização da Internet e da disponibilização de materiais, tais como livros e artigos *onlin*e, há uma ocorrência maior de cópias sem o devido uso de referências ao autor original. Neste trabalho, são abordadas técnicas de comparação de textos e formas de cálculo da similaridade entre dois blocos de arquivos, como o coeficiente de Jaccard e a similaridade dos cossenos. Combinando e adaptando algumas destas técnicas e formas, desenvolveu-se uma aplicação em Java que, com o uso da biblioteca Apache POI, captura o conteúdo de arquivos do Microsoft Word, divide o texto em blocos e consulta os blocos no Google, capturando todo o conteúdo HTML da página de consulta pela biblioteca Jsoup, e então compara os blocos calculando o coeficiente de Jaccard, permitindo detectar possíveis indícios de plágio, de uma forma mais eficiente que as encontradas nos *softwares* existentes.

Palavras-chave: Coeficiente de Jaccard; cópia; similaridade.

ABSTRACT

Because of the increasing use of the Internet and the availability of materials, such as books and papers online, making copies without acknowledging the original author of a text has become a recurrent practice. This paper aims to analyze techniques adopted to compare texts and methods to calculate the similarity between two blocks of files, i.e. the Jaccard index (aka the Jaccard similarity coefficient) and the cosine similarity measure. By combining and adapting some of these techniques, it was expected to develop a Java application using Apache POI library, which would capture contents from Microsoft Word files, divide the text in blocks and then search them on Google, thus capturing the HTML content from the page searched through Jsoup library. Finally, the blocks calculating the Jaccard similarity coefficient are compared to detect possible plagiarism in a more efficient way than those found in existing software.

Keywords: *The Jaccard Similarity Coefficient; copying; similarity.*

[1] Trabalho Final de Graduação - TFG.

[2] Acadêmica do Curso de Sistemas de Informação - Centro Universitário Franciscano. E-mail: caroldias.sm@gmail.com

[3] Orientador - Centro Universitário Franciscano. E-mail: sylviovieira@gmail.com

INTRODUÇÃO

A Internet e a globalização permitiram agilizar o cotidiano das pessoas. Pelo seu uso, é possível encontrar livros, artigos e debates sobre diversos assuntos, divulgando o conhecimento e aprendizado destes. Porém, essa facilidade de acesso também promove uma maior permissividade para cópias, que, em muitos casos, ocorre sem o uso das devidas referências as suas fontes.

Embora a legislação brasileira proíba a cópia sem autorização de livros e artigos, pesquisas comprovam que o número de trabalhos plagiados triplicou desde 1970 (AGÊNCIA BRASIL, 2011). Nota-se ainda uma falta de conscientização e de medidas que auxiliem na prevenção e detecção de tal infração. Apesar de já existirem ferramentas capazes de detectar plágio automaticamente, estas ferramentas ainda não apresentam resultados satisfatórios, deixando de identificar possíveis casos de plágio. Diante disto, faz-se necessária a existência de uma ferramenta que permita uma maior detecção, trazendo resultados mais coerentes. Protegendo e alertando a comunidade acadêmica de sua ocorrência, mostra assim uma necessidade de impetrar um cuidado maior, por parte dos alunos e professores, com aquilo que está sendo escrito.

Esta necessidade despertou o interesse de se desenvolver uma aplicação em Java. Por meio de técnicas de mineração de texto, como as descritas no trabalho de Zhang et al. (2013), é possível detectar trechos de trabalhos acadêmicos com um certo grau de similaridade com textos distribuídos pela Internet, caracterizando o plágio. Dessa forma, pode ser possível além de identificar, conscientizar e diminuir a ocorrência deste.

O objetivo deste trabalho foi desenvolver uma aplicação que, por meio de mineração de textos, auxilie na detecção de plágio acadêmico.

REFERENCIAL TEÓRICO

Nesta seção, são explicados conceitos e técnicas pertinentes para o desenvolvimento e entendimento deste trabalho.

PLÁGIO

Segundo Fachini e Domingues (2008), a reprodução indevida, seja ela total ou parcial, se configura como plágio. Ou seja, plágio não é somente a cópia integral de uma obra. Uma simples frase de outro autor, sem os devidos créditos já é considerado plágio. Existem três tipos de plágio: integral, parcial e conceitual, onde, neste último, apenas a essência da obra é copiada (GARSCHAGEN, 2006).

Embora a utilização deste termo com este sentido seja iniciada em cerca de 104 d.C. (CHRISTOFE, 1996), com as facilidades de acesso a mídias digitais oferecidas pela internet o

plágio não somente aumentou como permite que sejam copiados trabalhos baseados em outros trabalhos, o que dificulta a sua identificação.

No Brasil, o assunto é abordado oficialmente na lei que rege os Direitos Autorais, a lei 9.610 de 1998, e no Código Penal vigente com pena de detenção de 3 meses a 1 ano e multa.

MINERAÇÃO DE DADOS

Mineração de dados é o processo de exploração e análise de uma grande quantidade de dados com a finalidade de encontrar padrões e extrair novas informações (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). A mineração de dados se insere em diferentes áreas, como administração ou marketing, que a consideram como um importante recurso para estratégias de negócios. O uso da descoberta de informação através da mineração pode ser necessário para a competitividade no atual mundo dos negócios. Minerando os dados dos clientes, as empresas podem customizar melhor seus serviços e produtos, promovendo um produto ou serviço final mais eficiente. Porém, o uso de mineração de dados não é restrito à área de negócios.

A mineração de dados pode ser utilizada com diversas finalidades, como análise de performance de atletas e times descobrindo como melhorá-las, pesquisa de genes e DNA para a descoberta de diferentes tipos de cânceres e outras doenças, etc.

Minerar dados é um processo complexo, e, de acordo com a metodologia iterativa chamada Processo Padrão Inter-Indústrias para Mineração de Dados (em inglês *Cross Industry Standard Process for Data Mining* - CRISP-DM) é dividida nas seguintes etapas (KANTARDZIC, 2011; ORACLE, 2005):

- Entendimento do negócio: Nesta etapa, o especialista em mineração de dados e o especialista do negócio trabalham juntos para definir o objetivo do processo e planejar o projeto.
- Entendimento dos dados: Nesta etapa, o especialista de mineração coleta os dados e executa algumas atividades para se familiarizar com eles e definir quais são realmente necessários. A partir desta etapa, somente o especialista de dados desenvolve o processo.
- Preparação dos dados: Nesse estágio, são feitas a exclusão dos dados sem utilidade para o projeto e a seleção e conversão dos dados utilizados para um modelo adequado.
- Modelagem de dados: Essa etapa consiste na divisão dos dados em teste e treinamento para a execução de diversas técnicas (algoritmos). Cada algoritmo pode conter diferentes parâmetros, sendo necessário voltar a etapa de preparação dos dados para mudar o formato dos dados.
- Avaliação: Nesta fase. é feita a avaliação do modelo e resultados gerados com o especialista de negócio, comparando com os objetivos definidos
- Desenvolvimento: Essa fase engloba a apresentação dos resultados finais de uma forma clara para o cliente.

MINERAÇÃO DE TEXTO

A mineração de dados trabalha principalmente com dados estruturados em um SGBD. Porém, textos não são estruturados ou, em alguns casos, são semiestruturados, o que dificulta a manipulação e análise deste tipo de dado. Para melhor minerar textos, o processo de mineração foi adaptado e é dividido nas seguintes etapas: preparação dos dados, processamento dos dados e apresentação dos resultados (CORRÊA et al., 2012)

Preparação dos dados

Nesta etapa, são usadas técnicas para eliminar dados inexpressivos, que não auxiliarão posteriormente no processo de descoberta de informação, e selecionar apenas dados relevantes ao processo.

Algumas técnicas mais comumente utilizadas na preparação dos dados são:

- *Stopwords*: Uma lista de palavras que não acrescentam significado ao texto é criada e eliminada. São geralmente palavras de ligações, como preposições, artigos, pronomes e advérbios (MARCACINI et al., 2011).
- *Stemming*: Uma palavra pode ter variações de acordo com seu sufixo ou prefixo. Por exemplo, o verbo "acabar" pode ser escrito como "acabou", "acabei", dentre outras formas, dificultando o processo de comparação. Para diminuir as palavras que serão armazenadas existe a técnica de *Stemming*, que reduz as palavras do texto aos seus radicais. Porém, como muitas palavras serão reduzidas aos seus radicais, essa técnica reduz também a precisão de buscas futuras, já que só será possível buscar os radicais das palavras (MORAIS; AMBROSIO, 2007).
- *Thesaurus*: Esta técnica consiste em armazenar um dicionário de sinônimos para as palavras do texto.

Essas técnicas podem ser utilizadas sozinhas ou em conjunto, dependendo do objetivo do processo. Neste trabalho, as *stopwords* serão removidas, reduzindo sensivelmente as palavras significativas.

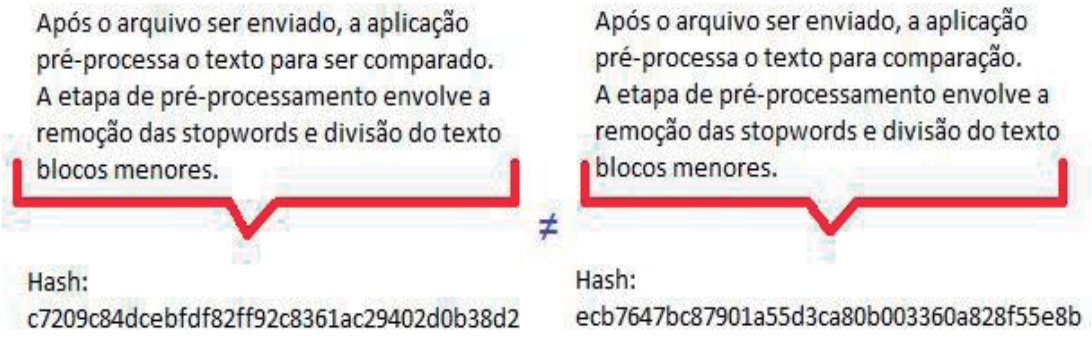
Processamento dos dados

Onde ocorre a extração de padrões. Existem diversas técnicas que podem ser usadas nessa etapa. Muitas delas têm o objetivo de agrupar os textos e diminuir sua quantidade em um documento sem perder o seu significado (REZENDE, 2003). Entretanto, como o escopo do trabalho é identificar o grau de similaridade entre textos, somente serão explicadas técnicas para este fim que, de acordo com Zhang et al. (2013), podem ser:

Algoritmo de *hashing* de todo o arquivo: Nessa técnica, cada texto tem apenas um valor *hash*[4] calculado de todo o seu conteúdo. Por meio deste algoritmo, só é possível verificar quando dois textos são totalmente idênticos.

Na figura 1 mostra-se um exemplo deste método, em que apenas uma pequena parte do texto foi modificada e, como os valores *hash* são diferentes, os textos são considerados totalmente diferentes.

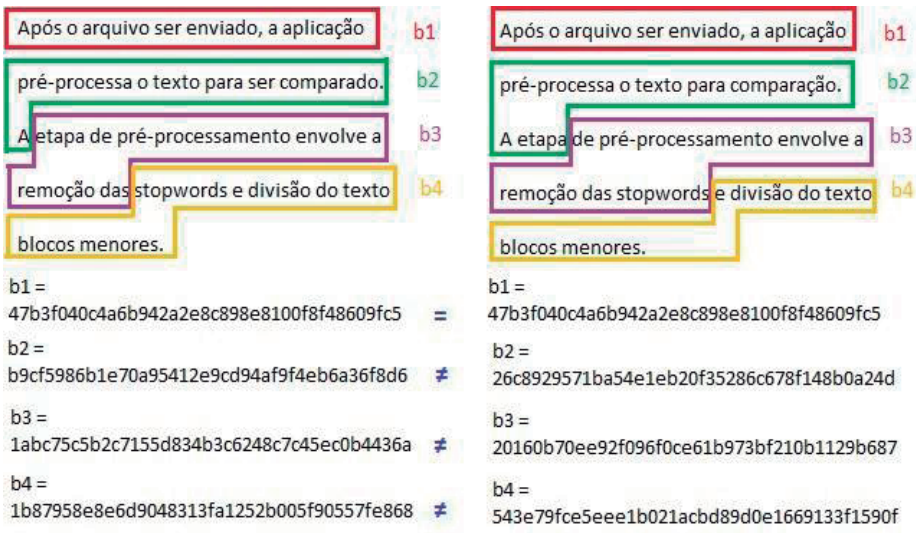
Figura 1 - Exemplo de Hashing de Todo Arquivo.



Fonte: elaborada pela autora.

Método de blocos de tamanho fixo: Nessa técnica, o texto é divido em blocos de tamanho fixo e, para cada bloco, é calculado um valor *hash*. Quando um valor *hash* dos blocos do texto é igual ao *hash* de um bloco de outro texto, os blocos são iguais. A desvantagem deste método é que, se um caractere qualquer do bloco for deletado ou alterado, além do valor *hash* do bloco ser diferente, isso também afetará os blocos seguintes, como pode ser visto na figura 2.

Figura 2 - Exemplo do Método de Blocos de Tamanho Fixo**.**

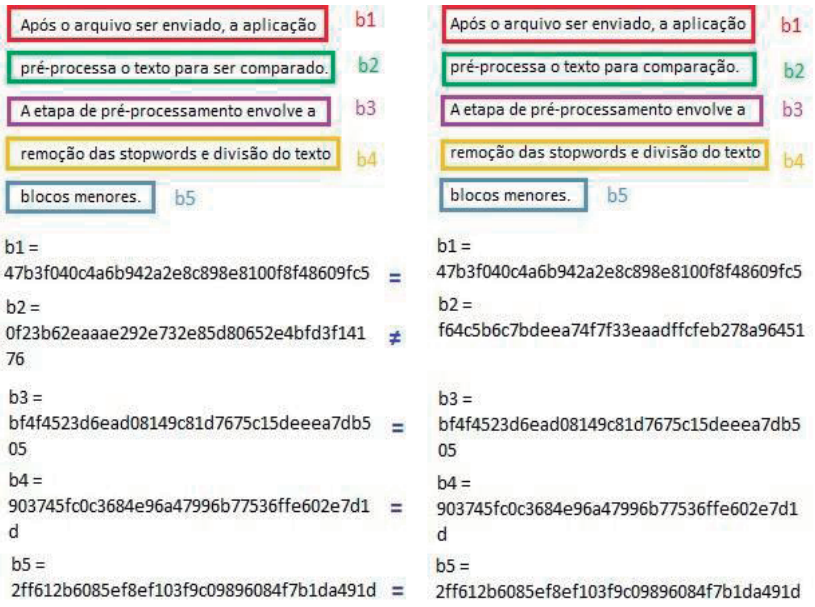


Fonte: elaborada pela autora.

[4] Funções *hash* são algoritmos que calculam um valor fixo para arquivos de tamanhos variáveis (AUMASSON et al., 2014). Geralmente utilizadas na verificação de integridade e autenticidade de arquivos.

Método de blocos por conteúdo: Neste algoritmo, os blocos não possuem tamanho fixo, mas são divididos pelo seu conteúdo, por exemplo, por linha. A vantagem desta técnica é que, se algum caractere de um bloco for alterado ou deletado, apenas alguns blocos sofrerão alteração.

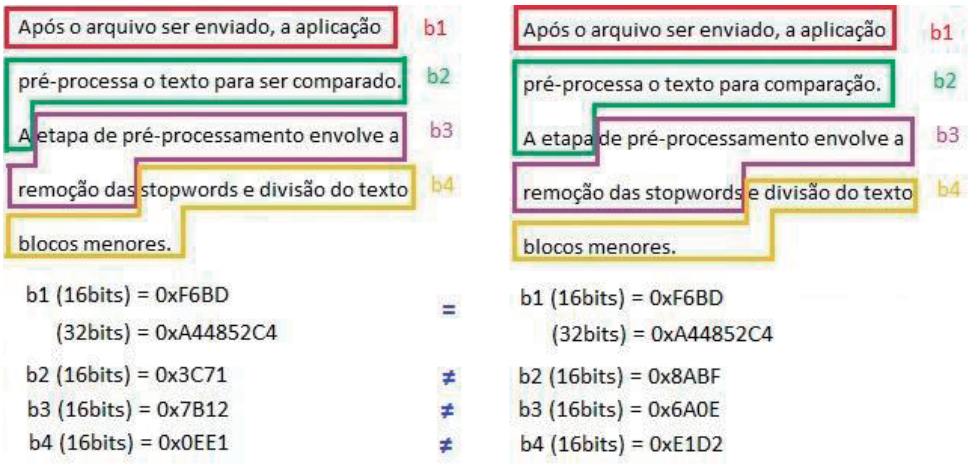
Figura 3 - Exemplo do Método de Blocos por Conteúdo.



Fonte: elaborada pela autora.

Método do bloco deslizante: Esse método também divide o texto em blocos fixos mantendo uma soma de verificação[5] forte e uma fraca. Quando um novo texto é verificado, se a comparação das somas de verificação fraca for igual, a soma de verificação forte é calculada e comparada. Se a soma de verificação forte também for igual, então o bloco é igual, como pode ser visto na figura 4.

Figura 4 - Exemplo do Método do Bloco Deslizante.



Fonte: elaborada pela autora.

[5] Soma de verificação (do inglês *checksum*) é a soma das palavras de um arquivo. É utilizada na verificação da sua integridade. Existem dois tipos, a fraca e o forte. A fraca possui 16 bits e a forte é a partir de 32 bits (JONES, 2010; PETERSON; DAVIE, 2013).

A aplicação detectará não somente blocos iguais, como blocos que tem um alto grau de similaridade. Será usado então, o método de blocos de tamanho fixo e, ao invés de comparar pelo valor *hash,* os blocos serão comparados palavra a palavra. Dessa forma, mesmo se for alterado algum caractere no meio do bloco, ainda será possível detectar que algumas palavras do bloco são iguais.

Apresentação dos Resultados

Nesta etapa, os resultados do processo de mineração de texto são apresentados e avaliados.

COEFICIENTE DE JACCARD E SIMILARIDADE DO COSSENO

Há duas fórmulas matemáticas principais para determinar a similaridade entre dois conjuntos, o coeficiente de Jaccard e a Similaridade do Cosseno.

O coeficiente de Jaccard é determinado pela relação da intersecção de dois conjuntos com a união destes (SILVESTRE, 2009), como demonstra a fórmula:

$$Cj = \frac{a}{a+b+c} \tag{1}$$

sendo:

a = o número de elementos em comum dos dois conjuntos;

b = o número de elementos presentes apenas no conjunto 1;

c = o número de elementos presentes apenas no conjunto 2.

Dados dois vetores, a similaridade do Cosseno é determinada pelo ângulo entre estes dois vetores (CORREA et al., 2012), como demonstra a fórmula:

$$\cos(x,y) = \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{(\sum(x_i - \bar{x})^2)\,(\sum(y_i - \bar{y})^2))}} \tag{2}$$

sendo:

x_i e y_i = os valores de cada termo;

$\bar{x}$ e $\bar{y}$ = as médias das variáveis.

Ambas as fórmulas retornam valores entre 0 e 1. Quanto mais próximo de 1, maior a similaridade dos conjuntos.

Neste trabalho, decidiu-se calcular a similaridade dos blocos por meio do coeficiente de Jaccard, por ser um dos mais utilizados (BARROS, 2007) e por melhor se adequar à implementação da aplicação.

TRABALHOS CORRELATOS

Nesta seção, são demonstrados alguns trabalhos com características similares ou que de alguma forma contribuíram para o desenvolvimento deste trabalho.

Detecção de duplicatas para reduzir o armazenamento e o consumo de banda

Zhang et al. (2013) descreveram as técnicas existentes de comparação de textos, como o método de blocos de tamanho fixo e o por conteúdo. Além disso, relatam no artigo uma nova técnica, mesclando o método do bloco deslizante com indexação. O método do bloco deslizante é o mais eficiente em termos de espaço de armazenamento. Porém, quando se está utilizando um servidor para armazenar os arquivos, ainda é preciso enviar todo o texto ao servidor para o cálculo da soma de verificação, o que consome um grande tráfego de dados.

O sistema proposto no artigo aprimora essa técnica, calculando a soma de verificação dos blocos e os enviando antes do envio do arquivo, para serem comparados com as somas de verificação dos arquivos que já estão no servidor. O sistema também possui um índice com os endereços dos blocos. Assim, cada vez que um bloco for igual ao outro, é preciso apenas gravar um ponteiro indicando onde o bloco está.

Detecção automática de plágio usando análise de similaridade

Hariharan (2012) relata um estudo realizado com estudantes de uma universidade. Os estudantes foram divididos em pequenos grupos e cada grupo deveria escrever uma redação com tema diferente em um período de uma semana.

As redações dos grupos foram pré-processadas (utilizando as técnicas de *stopwords* e *stemming*). Por meio de fórmulas de análise de similaridade, tais como coeficiente de Jaccard e similaridade dos cossenos, uma grande quantidade de plágio foi detectada.

Quando as mesmas redações foram analisadas utilizando softwares disponíveis *online* (tais como ArticleChecker, Viper e Duplichecker), notou-se que estas detectaram menos ou não detectaram nenhum plágio nos trabalhos.

Considerações

Os artigos correlatos contribuem para o desenvolvimento deste trabalho, pois cada um foca em uma parte diferente do processo de detecção de cópias de texto. Zhang et al. (2013) foca nas técnicas de divisão de texto e no impacto que estas técnicas têm na detecção de cópias. Hariharan (2012) demonstra que os *softwares* disponíveis ainda não são muito eficientes na detecção de plágio, reportando uma necessidade de aprimoramento destes softwares utilizando técnicas de análise de similaridade.

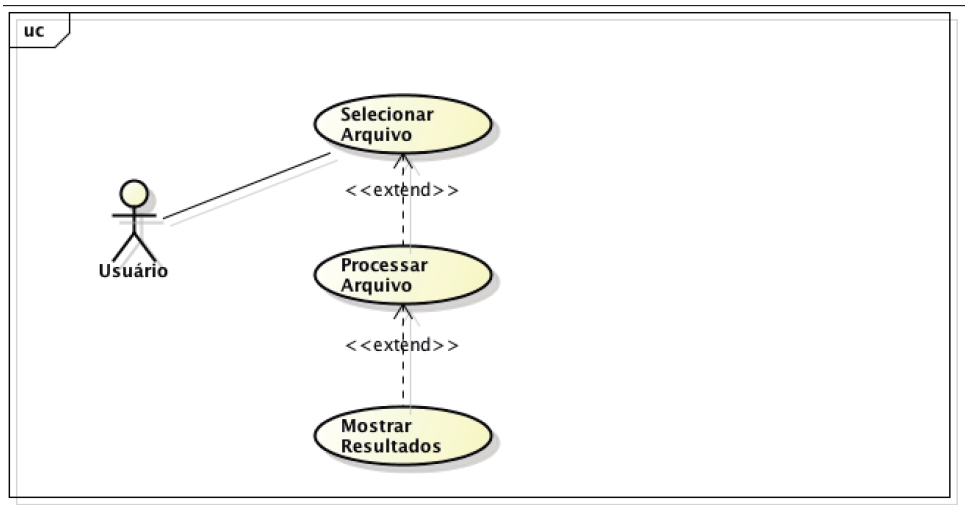
MÉTODOS

Para desenvolver este trabalho, foram utilizadas a metodologia de mineração de textos, a linguagem Java e as bibliotecas Apache POI, para capturar o conteúdo de arquivos Microsoft Word, e Jsoup, para capturar o conteúdo HTML das páginas de consulta.

Combinando e adaptando o método do bloco de tamanho fixo e o coeficiente de Jaccard, a aplicação proposta detecta possíveis plágios da seguinte forma:

O usuário pode selecionar e processar um arquivo no formato do Microsoft Word. As ações do usuário são demonstradas no Diagrama de Caso de Uso, que pode ser visualizado na figura 5.

Figura 5 - Diagrama de Caso de Uso.



Fonte: elaborada pela autora.

Como pode ser visto no Diagrama de Atividade da figura 6, após o arquivo ser enviado, a aplicação pré-processa o texto para ser comparado. A etapa de pré-processamento envolve a remoção de acentos e letras, conversão de todas as letras para minúsculo, a remoção das *stopwords*, lista de acordo com Linguateca (2006), e a divisão do texto em blocos menores. Porém, não são calculados os valores *hash* de cada bloco, como explicado em Zhang et al. (2013). Com os blocos divididos, é feita uma consulta ao Google. Cada resultado da consulta é interpretado como um bloco e é pré-processado da mesma forma que o texto do usuário. As palavras de ambos os blocos (pesquisado e resultante) são divididas e armazenadas em vetores (um vetor para palavras do bloco pesquisado e um vetor para palavras do bloco resultante).

Para calcular a similaridade de cada bloco, as palavras repetidas são eliminadas e, após isso, é utilizado o coeficiente de Jaccard, explicado em Hariharan (2012). Para isso, contadores são utilizados de acordo com as condições:

- Se uma palavra de um bloco resultante é igual a uma palavra do bloco pesquisado, é adicionado 1 ao contador de elementos presentes em ambos os blocos.
- O contador de elementos presentes apenas no bloco pesquisado recebe o valor do número de palavras do bloco pesquisado menos o número de elementos em comum aos dois blocos.
- O contador de elementos presentes apenas no bloco resultante recebe o valor do número de palavras do bloco resultante menos o número de elementos em comum aos dois blocos.

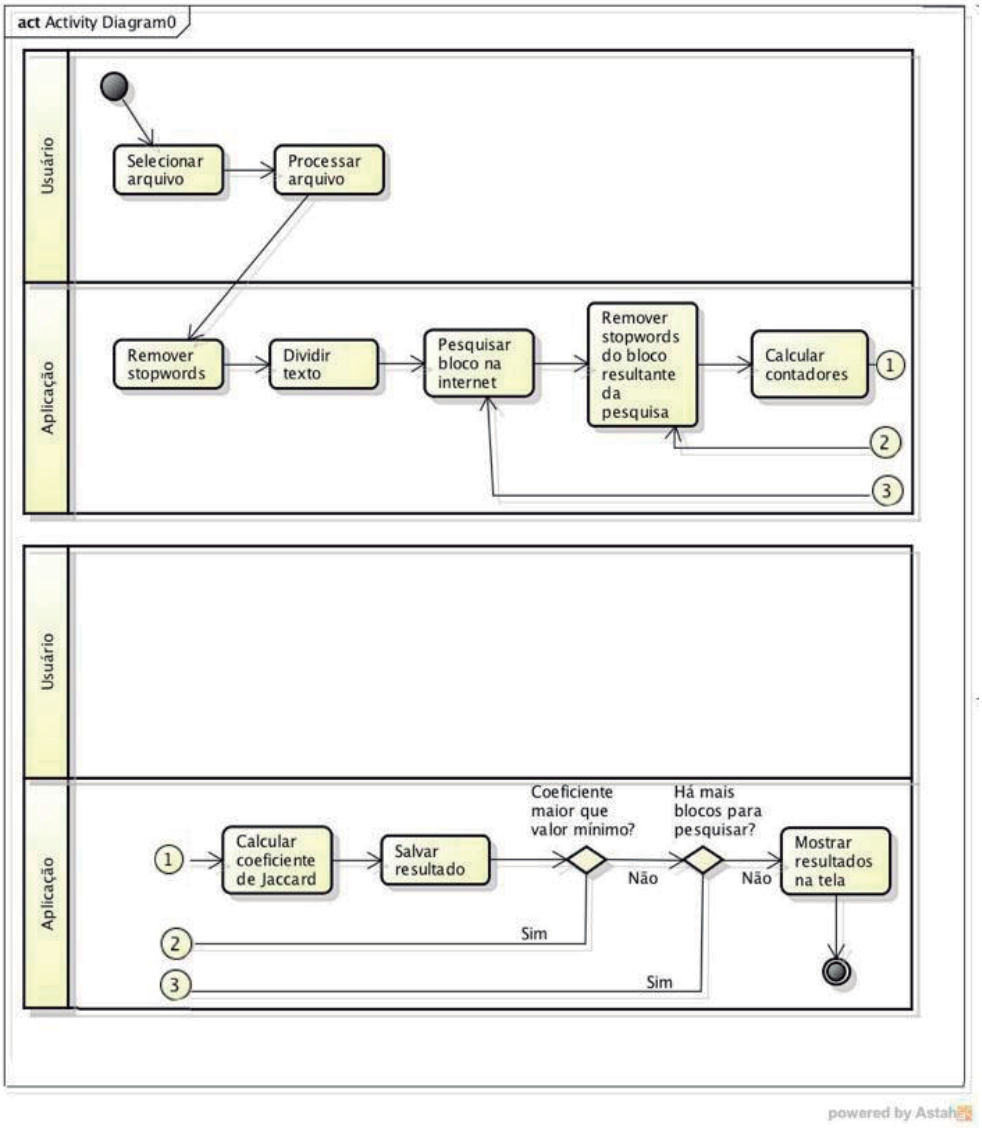
Após as palavras serem armazenadas em vetores, a fórmula do coeficiente de Jaccard é calculada com a utilização dos contadores. Se o resultado for maior que 0,5, pois foi verificado através de testes que é um índice que traz resultados mais coerentes, o bloco é armazenado e apresentado ao usuário após a verificação dos blocos.

Na figura 7, mostra-se um exemplo de bloco pesquisado e resultante e como os contadores e o cálculo do coeficiente de Jaccard é feito.

Um detalhe importante a ser notado é que podem existir milhares de resultados de pesquisa. Se a aplicação for analisar todos eles, ficará extremamente lenta e poderá acarretar outros problemas, como, por exemplo, utilizar toda a memória do computador do usuário. Para evitar esse tipo de problema, sabendo que o Google mostra os resultados da pesquisa em ordem decrescente de relevância, a aplicação verifica os resultados até que o coeficiente de Jaccard seja menor a 0,5, ou os dez primeiros resultados.

No final, a lista de resultados relevantes, ou seja, que tem grande probabilidade de se configurar como plágio, são mostrados ao usuário para análise humana e tomada de decisão.

Figura 6 - Diagrama de Atividade.



Fonte: elaborada pela autora.

Figura 7 - Exemplo de contadores e cálculo do Coeficiente de Jaccard.

BLOCO PESQUISADO:
trechos principe desenho seguramente menos sedutor modelo tenho culpa fora desencorajado seis anos carreira pintor aprendera desenhar jiboias abertas fechadas olhava aparicao olhos redondos espanto esquecam achava mil milhas qualquer terra habitada ora homenzinho parecia perdido morto fadiga morto fome sede medo absolutamente aparencia crianca perdida deserto mil milhas regiao

BLOCO RESULTANTE:
desenha carneiro desenho seguramente menos sedutor modelo fora desencorajado seis anos carreira pintor aprendera olhava aparicao olhos redondos espanto esquecam achava mil milhas qualquer terra habitada

Debug da Aplicação:

cont	#3536(length=3)
a = [0]	26
b = [1]	24
c = [2]	0
coeficiente	0.52

Coeficiente de Jaccard

$$Cj = \frac{a}{a+b+c}$$

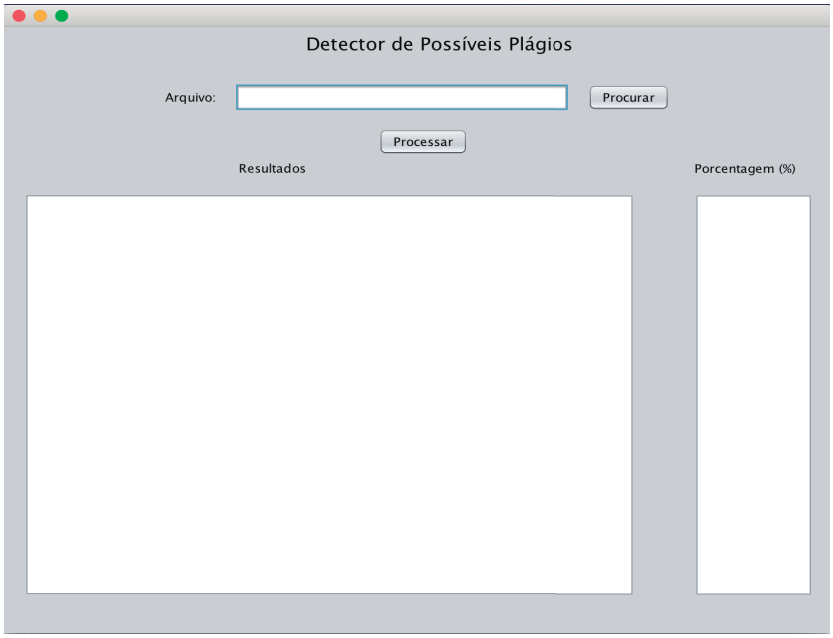
$$Cj = \frac{26}{26+24+0} = 0{,}52$$

Fonte: elaborada pela autora.

TELA

A figura 8 corresponde à tela da aplicação. Através dessa figura, é possível compreender que o usuário pode escolher o arquivo e processar. Os resultados da análise de similaridade são mostrados logo abaixo do botão de processamento do arquivo.

Figura 8 - Tela da Aplicação.



Fonte: elaborada pela autora.

RESULTADOS E DISCUSSÃO

Para demonstrar os resultados da aplicação, foi utilizado um arquivo de exemplo, contendo fragmentos do livro "O Pequeno Príncipe", intercalado com textos inseridos aleatoriamente, como pode ser observado na figura 9 - foi escolhido esse texto de exemplo por ser um livro muito difundido e, assim, pode-se ter certeza que a consulta retornará vários resultados, o que facilita a análise. Na figura 10, percebe-se que a aplicação encontra e exibe na tela um trecho com 52% de semelhança ao teste (com blocos divididos a cada 50 palavras). Neste caso, há a necessidade de intervenção humana na comparação do resultado com o texto original para verificar se este trecho possui alguma referência e se os sentidos das orações são realmente semelhantes.

Na figura 11, o mesmo texto foi pesquisado, porém com a divisão dos blocos a cada 25 palavras. Como pode ser notado, o valor do coeficiente e a quantidade de blocos aumentou. Pois cada bloco resultante tem, geralmente, 50 palavras (sem remover *stopwords*) e em média 25 palavras após a remoção das *stopwords*, tornando o tamanho do bloco resultante e do bloco pesquisado praticamente iguais. Assim, o valor do coeficiente se torna mais confiável.

Porém, o Google possui um sistema de bloqueio de pesquisas automatizadas, o que limita o uso da aplicação em cerca de 100 consultas diárias. Para aumentar o número de textos verificados, os blocos foram, então, limitados a cada 50 palavras. Dessa forma, é possível detectar possíveis plágios, embora com um coeficiente menor, e verificar mais textos.

Figura 9 - Arquivo Word de Teste.

Trechos de O Pequeno Príncipe

Meu desenho é, seguramente, muito menos sedutor que o modelo. Não tenho culpa. Fora desencorajado, aos seis anos, da minha carreira de pintor, e só aprendera a desenhar jibóias abertas e fechadas.

Olhava pois essa aparição com olhos redondos de espanto. Não esqueçam que eu me achava a mil milhas de qualquer terra habitada. Ora, o meu homenzinho não me parecia nem perdido, nem morto de fadiga, nem morto de fome, de sede ou de medo. Não tinha absolutamente a aparência de uma criança perdida no deserto, a mil milhas da região habitada. Quando pude enfim articular palavra, perguntei-lhe:

- Mas ... que fazes aqui?

E ele repetiu-me então, brandamente, como uma coisa muito séria:

- Por favor... desenha-me um carneiro ...
- Não tem importância. Desenha-me um carneiro.

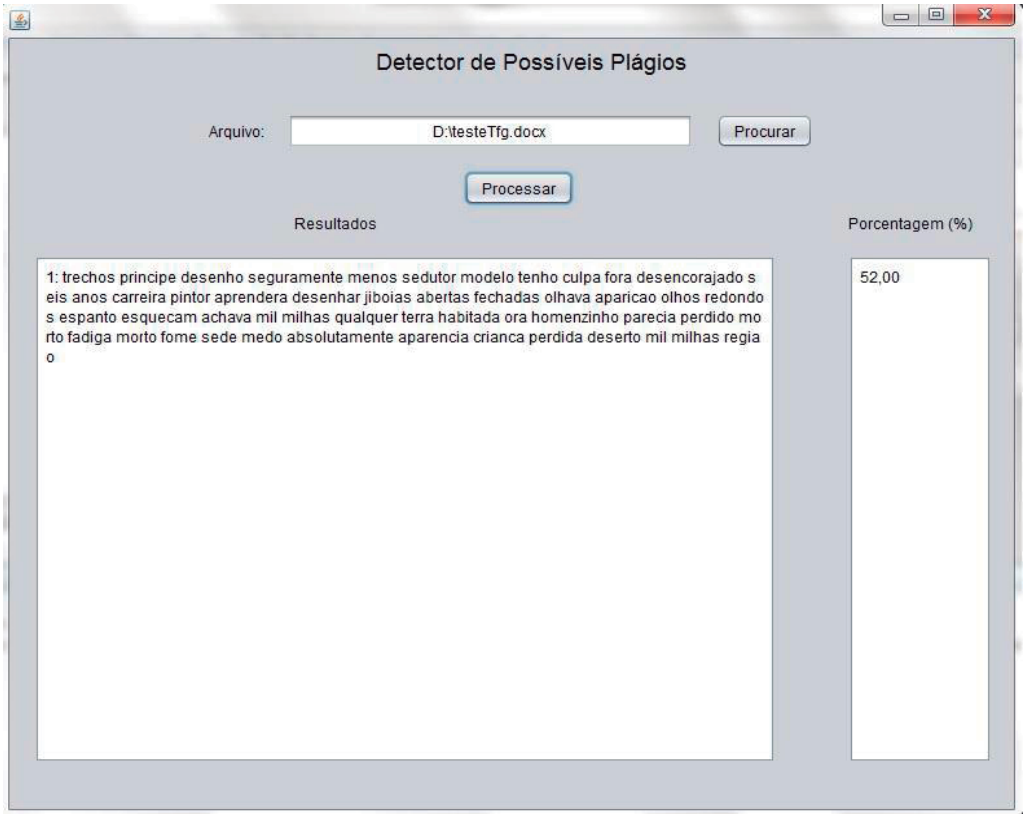
Trecho de teste do tfg. Verificando plágios.

Como jamais houvesse desenhado um carneiro, refiz para ele um dos dois únicos desenhos que sabia. O da jibóia fechada. E fiquei estupefato de ouvir o garoto replicar:

- Não! Não! Eu não quero um elefante numa jibóia. A jibóia é perigosa e o elefante toma muito espaço. Tudo é pequeno onde eu moro. Preciso é dum carneiro.

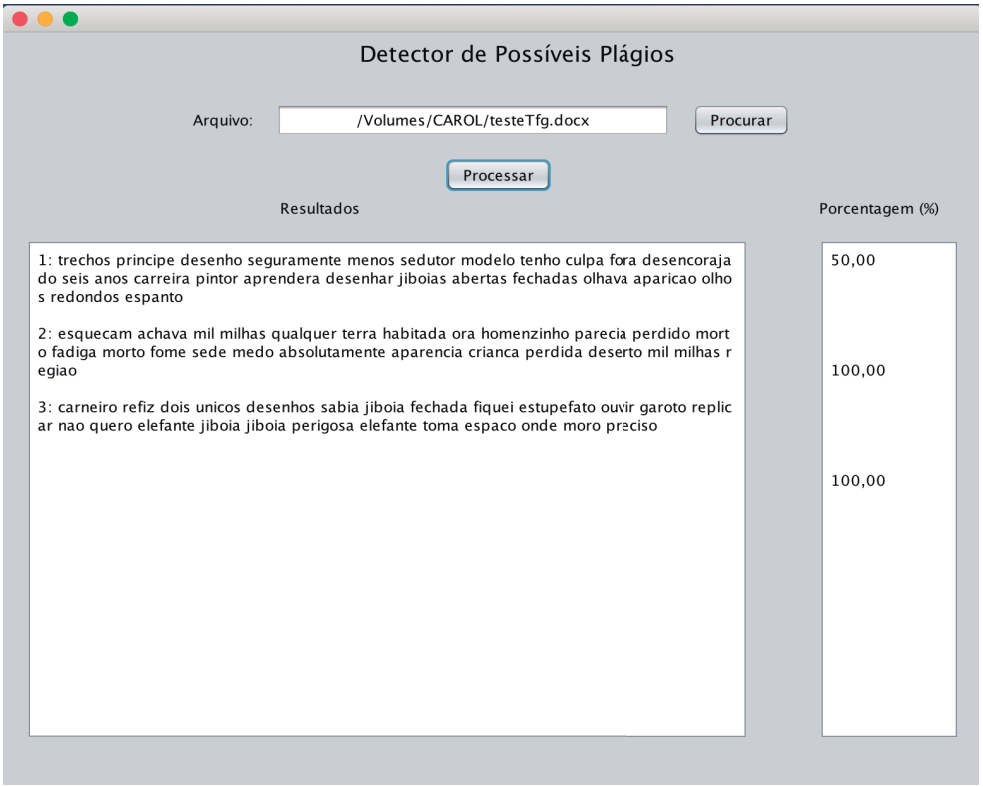
Fonte: elaborada pela autora.

Figura 10 - Tela de Resultado de Teste com Blocos de 50 Palavras.



Fonte: elaborada pela autora.

Figura 11 - Tela de Resultado de Teste com Blocos de 25 Palavras.



Fonte: elaborada pela autora.

Este trabalho contribui academicamente pela forma como foi conduzido, pela aplicação do coeficiente de Jaccard e pelos tamanhos de blocos utilizados. Desta forma, foi possível trazer não somente os trechos dos textos com similaridade, como seu percentual dentro do bloco. Para comparação de resultados, foram utilizadas as mesmas ferramentas que em Hariharan (2012), que apresentaram resultados não satisfatórios. Quando o texto foi comparado pela ArticleChecker, nenhuma similaridade foi encontrada. Já no caso da Duplichecker, foi retornado um texto com as palavras iguais destacadas. No caso da Viper, há um quesito anterior à busca que necessita ser configurado, identificando a área de atuação do texto. Como não havia a opção de livros ou trechos de livros, a ferramenta não retornou nenhuma similaridade nas áreas testadas. Como pode ser notado, os resultados apresentados pela ferramenta desenvolvida são mais coerentes, detectando possíveis plágios de uma forma mais confiável. Vale ressaltar que, como a aplicação compara palavra a palavra, não é detectado quando uma palavra é traduzida de outra língua. Porém, se o texto pesquisado possuir resultados de pesquisas do Google na mesma linguagem do pesquisado (não importando qual seja esta), a aplicação detectará que são textos semelhantes.

TRABALHOS FUTUROS

Muito ainda deve ser pesquisado e aplicado a este trabalho, como incluir outros motores de busca, para aumentar a área de abrangência e sanar as limitações que o Google impõe. Também pode-se utilizar sistemas distribuídos de forma que os blocos sejam pesquisados por diferentes máquinas com diferentes endereços, aumentando desta forma o número de consultas diárias possíveis.

REFERÊNCIAS

AGÊNCIA BRASIL. **Aumento do plágio em produções científicas preocupa pesquisadores em todo o mundo.** 2011. Disponível em: <https://goo.gl/reNYYQ>. Acesso em: 03 dez. 2015.

AUMASSON, J. P. et al. **The Hash Function BLAKE**. Berlin: Springer, 2014. p. 1.

BARROS, R. S. M. **Medidas de Diversidade Biológica**. Universidade Federal de Juiz de Fora, 2007. Disponível em: <https://goo.gl/6DNS95>. Acesso em: 03 dez. 2015.

CHRISTOFE, L. **Intertextualidade e plágio:** questões de linguagem e autoria. 1996. Disponível em: <https://goo.gl/o0M9Og>. Acesso em: 03 dez. 2015.

CORRÊA, G. N.; MARCACINI, R. M.; REZENDE, S. O. **Uso da mineração de textos na análise exploratória de artigos científicos**. 2012. Disponível em: <https://goo.gl/9Gl7Vj>. Acesso em: 03 dez. 2015.

FACHINI, G. J.; DOMINGUES, M. J. C. S. Percepção do Plágio Acadêmico entre Alunos de Programas de Pós-graduação em Administração e Contabilidade. In: XI SEMINÁRIOS EM ADMINISTRAÇÃO DA FEA-USP - SEMEAD, **Anais...** p. 1-14, 2008. São Paulo.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. **From Data Mining to Knowledge Discovery in Databases**. American Association for Artificial Intelligence, 1996.

GARSCHAGEN, B. **Universidade em tempos de plágio**. 2006. Disponível em: <https://goo.gl/A8f47t>. Acesso em: 03 dez. 2015.

HARIHARAN, S. Automatic Plagiarism Detection Using Similarity Analysis. **The International Arab Journal of Information Technology**, v. 9, n. 4, p. 322-326, 2012.

JONES, E. **More Than You Wanted to Know About Checksums**. 2010. Disponível em: <https://goo.gl/FkP99Y>. Acesso em: 06 jun. 2015.

KANTARDZIC, M. **Data Mining:** Concepts, Models, Methods, and Algorithms. Canada: Wiley, 2011.

LINGUATECA. **Lista de palavras frequentes em Português**. 2006. Disponível em: <https://goo.gl/f6Iovs>. Acesso em: 03 dez. 2015.

MARCACINI, R. M.; MOURA, M. F.; REZENDE, S. **O uso da Mineração de Textos para Extração e Organização Não Supervisionada de Conhecimento**. 2011. Disponível em: <https://goo.gl/g20Qb9>. Acesso em: 03 dez. 2015.

MORAIS, E. A. M.; AMBRÓSIO, A. P. L. **Mineração de Textos**. 2007. Disponível em: <https://goo.gl/htaMZv>. Acesso em: 03 dez. 2015.

ORACLE. **What is data mining**. 2005. Disponível em: <https://goo.gl/x6J395>. Acesso em: 03 dez. 2015.

PETERSON, L. L.; DAVIE, B. S. **Redes de Computadores:** Uma Abordagem de Sistemas. Rio de Janeiro: Elsevier, 2013. p. 59.

REZENDE, S. O. **Sistemas Inteligentes:** fundamentos e aplicações. Barueri: Manole, 2003.

SILVESTRE, R. **Comparação da Florística, Estrutura e Padrão Espacial em Três Fragmentos de Floresta Ombrófila Mista no Estado do Paraná**. 2009. Disponível em: <https://goo.gl/u1tnFs>. Acesso em: 03 dez. 2015.

ZHANG, J. et al. DDSN: Duplicate Detection to Reduce Both Storage and Bandwidth Consumption. IEEE International Conference on Big Data. **Anais...** Santa Clara, 2003. p. 206-211.