UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

SUMARIZANDO INFORMAÇÕES NA *WEB* ATRAVÉS DA IDENTIFICAÇÃO DE TENDÊNCIAS

TRABALHO DE GRADUAÇÃO

Vinicius Aquino Rodrigues

Santa Maria, RS, Brasil

2016

SUMARIZANDO INFORMAÇÕES NA *WEB* ATRAVÉS DA IDENTIFICAÇÃO DE TENDÊNCIAS

Vinicius Aquino Rodrigues

Trabalho de Graduação apresentado ao Curso de Ciência da Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de

Bacharel em Ciência da Computação

Orientador: Prof[a]. Dr. Sérgio Luís Sardi Mergen

419
Santa Maria, RS, Brasil

2016

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

SUMARIZANDO INFORMAÇÕES NA *WEB* ATRAVÉS DA IDENTIFICAÇÃO DE TENDÊNCIAS

elaborado por
Vinicius Aquino Rodrigues

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

Sérgio Luís Sardi Mergen, Dr.
(Presidente/Orientador)

Marcia Pasin, Dra. (UFSM)

Giovani Rubert Librelotto, Dr. (UFSM)

Santa Maria, 14 de Dezembro de 2016.

À minha mãe Marli e à minha tia Luci

AGRADECIMENTOS

Obrigado ao meu orientador e a minha banca.

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

SUMARIZANDO INFORMAÇÕES NA *WEB* ATRAVÉS DA IDENTIFICAÇÃO DE TENDÊNCIAS

AUTOR: VINICIUS AQUINO RODRIGUES
ORIENTADOR: SÉRGIO LUÍS SARDI MERGEN
Local da Defesa e Data: Santa Maria, 14 de Dezembro de 2016.

Uma área de pesquisa que vem conquistado espaço é a descoberta de tendências a partir de dados disponibilizados na *Web*, ou seja, a descoberta de assuntos que despertem maior atenção por parte de usuários *Web*. Alguns trabalhos nessa área realizam o processo de descoberta usando técnicas relativamente complexas baseadas em aprendizado de máquina ou usando um volume grande de dados obtidos de bases de dados internas. O objetivo deste trabalho é demonstrar que é possível descobrir assuntos de pertinência através de implementações simples, analisando uma quantidade baixa de informações. Para isso, o trabalho propõe uma ferramenta chamada SE Trends (Search Engine Trends), que delega a parte mais onerosa do processamento a um motor de busca. A ferramenta aplica técnicas de recuperação de informação sobre páginas retornadas a partir de consultas efetuadas em um motor de busca. Cada termo encontrado é classificado quanto à sua importância, e termos mais bem classificados são considerados os assuntos mais relevantes. Os experimentos demonstram cenários em que o SE Trends retorna termos que realmente estão associados a assuntos de relevância nacional.

Palavras-chave: Recuperação de informação. palavras tendência. parsing. processamento de texto. nuvem de palavras.

ABSTRACT

Undergraduate Final Work
Computer Science
Federal University of Santa Maria

SUMMARIZING INFORMATION ON WEB BY TRENDING IDENTIFICATION

AUTHOR: VINICIUS AQUINO RODRIGUES
ADVISOR: SÉRGIO LUÍS SARDI MERGEN
Defense Place and Date: Santa Maria, December 14th, 2016.

A research area that has been gaining space is the discovery of trending topics from data published on the Web. Some works perform the discovery process using relatively complex algorithms based on machine learning or using large volume of data gathered from internal databases. The purpose of this work is to show that it is possible to discover relevant topics through simple strategies and analyzing a small amount of information. To achieve this goal, we propose a tool called SE Trends (Search Engine Trends), which delegates the time consuming processing to a search engine. The tool applies information recovery techniques over pages returned from queries submitted to a search engine. Each term found is classified according to its importance, and the best classified terms are considered the most important topics. The experiments show scenarios where SE Trends returns terms that are indeed associated with issues of national concern.

Keywords: information retrieval, trending words, parsing, text processing, word clouds.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
DOM	Document Object Model
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
LSI	Latent Semantic Indexing
PHP	PHP: Hypertext Preprocessor
SAX	Simple API for XML
SVM	Support Vector Machine
TF-IDF	Term Frequency–Inverse Document Frequency
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	Extensible Markup Language

SUMÁRIO

1 INTRODUÇÃO

Vive-se uma época em que pode-se acessar a uma grande quantidade de informações geradas na *Web*. Para se ter uma ideia, o serviço chamado *Internet On Real Time* estima que sejam gerados aproximadamente 1 *terabyte* de dados a cada minuto (LAB, 2016). Esse grande volume de dados é rico em conhecimento. No entanto, o volume é tão alto que dificulta a localização e/ou sumarização desse conhecimento de forma manual. Ou seja, um usuário não tem condições de acessar todas informações disponibilizadas para construir esse conhecimento.

Dentro desse contexto, uma área de pesquisa interessante é a identificação de tendências, ou seja, assuntos que despertem o interesse em usuários *Web*. Esse tipo de assunto pode ser encontrado em textos publicados em mídias como redes sociais, blogs e sites de notícia. Por exemplo, na rede social *Twitter*, os assuntos que predominam são denominados *trending topics*, e são recuperados a partir de informações contidas nos *tweets*. Já o serviço da *Google*, chamado de *Google Trends*, infere tendências a partir dos termos mais pesquisados pelos usuários do motor de busca. Em ambos os casos, a identificação de tendências depende do acesso a um volume de dados considerável, e o acesso completo é restrito à própria empresa, que possui essa informação em seus repositórios internos.

Além disso, a análise de tendências, mesmo em bases menores, costuma ficar a cargo de técnicas computacionais sofisticadas. Técnicas que são bastante utilizadas para esse fim são aquelas baseadas em aprendizado de máquina. Algoritmos dessa categoria podem ser usados, por exemplo, quando o objetivo é classificar documentos de acordo com o tópico. Nesse caso, técnicas de aprendizado não supervisionado poderiam ser usadas. Já as técnicas supervisionadas seriam adequadas para outras finalidades, como por exemplo, quando o interesse é realizar previsões à respeito de quais tópicos têm potencial para se tornar tendências no futuro.

No trabalho aqui proposto, o objetivo é demonstrar que certos tipos de resultado podem ser gerados utilizando técnicas simples de recuperação de informação, sem precisar analisar volumes muito grandes de dados. Para isso, o trabalho propõe uma ferramenta, chamada de SE Trends (Search Engine Trends). A ferramenta parte do pressuposto que assuntos pertinentes são mencionados em páginas *Web* de alta relevância. Assim, é possível delegar a um motor de busca a responsabilidade de recuperar essas páginas. A partir de uma lista de consultas (referentes a temas que se deseja pesquisar), a ferramenta analisa os documentos retornados pelo motor de busca e classifica os termos contidos nesses documentos. Essa análise é realizada

a partir de técnicas de processamento de texto e indexação. Os termos melhor classificados são considerados como os assuntos mais relevantes aos temas fornecidos pelo usuário. Além disso, os assuntos relevantes são apresentados na forma de uma nuvem de palavras, com o intuito de gerar uma visualização que sumarize o interesse despertado por um tema em particular.

O texto do trabalho está organizado da seguinte forma: No capítulo 2 é apresentada a fundamentação teórica que explica os conceitos cuja compreensão foi necessária para o desenvolvimento do trabalho. No Capítulo 3 são apresentados trabalhos relacionados à obtenção, tratamento e representação de tendências. No Capítulo 4 é apresentada a arquitetura da solução proposta. Nesse capítulo todas as etapas de processamento são descritas, desde a obtenção dos dados até a sua formatação para exibição. No Capítulo 5 são apresentados os experimentos realizados a partir da ferramenta desenvolvida. As conclusões são apresentadas no Capítulo 6.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta conceitos e fundamentos que são pertinentes no contexto deste trabalho de graduação. Os conceitos estão relacionados aos objetivos do trabalho, que envolvem a identificação de documentos e o seu processamento.

2.1 Recuperação de Informação

Recuperação de informação é a atividade de obter informações relevantes a partir de uma coleção de fontes de informação. Um processo de recuperação de informação ocorre tipicamente quando um usuário envia uma consulta a um sistema, como por exemplo uma busca em um motor de busca na *Web*. Uma consulta pode corresponder a vários objetos de uma determinada coleção e esses objetos podem ter diferentes graus de relevância.

Um objeto é uma entidade que armazenada em um repositório. Dependendo da aplicação, os objetos podem ser documentos de texto, páginas HTML, imagens (GOODRUM, 2000), áudios (FOOTE, 1999) ou vídeos.

Na recuperação de informação os resultados retornados podem ou não corresponder à consulta, de maneira que os resultados são tipicamente ranqueados. O ranqueamento de resultados é um dos fatores que torna a recuperação de informação uma atividade diferente da recuperação de dados, que é comumente usada em sistemas de banco de dados tradicionais, como os bancos relacionais(JANSEN; RIEH, 2010).

Sistemas de recuperação de informação costumam computar um valor numérico de acordo com a correspondência entre o objeto e a consulta, fazendo um ranqueamento de acordo com o valor. Os objetos melhores ranqueados são mostrados ao usuário e o processo pode ser iterado caso o usuário queira refinar a consulta (FRAKES, 1992).

Nesse contexto, são particularmente importante os conceitos de preprocessamento dos documentos, indexação de documentos e estratégias de ranqueamento. Cada um desses conceitos será discutido nas seções que seguem, a partir da perspectiva de consultas em motores de busca, que é foco deste trabalho.

2.1.1 Preprocessamento de documentos

O preprocessamento de texto é uma etapa aplicada sobre um documento de forma a transformá-lo em uma versão mais aproveitável para alguma finalidade específica. Essa finalidade pode estar relacionada, por exemplo, ao armazenamento do documento em um repositório. Para o caso específico de motores de busca que provém acesso à páginas HTML, o preprocessamento visa adequar o documento em função dos termos que costumam ser usados em consultas. Algumas técnicas de preprocessamento úteis são a remoção de *stop words* e a aplicação de *stemming*.

Stop words são palavras que podem ser removidas do documento sem que haja prejuízo no processamento posterior. (LESKOVEC; RAJARAMAN; ULLMAN, 2014). A lista de palavras vazias pode variar de acordo com o objetivo da aplicação, sendo comum a inclusão de palavras comuns e palavras funcionais, como por exemplo artigos, preposições e conjunções.

Já a aplicação de *Stemming* tem como propósito reduzir flexões e formas derivadas de uma palavra para uma forma base comum (MANNING; RAGHAVAN; SCHüTZE, 2009). Para isso, o *stemming* utiliza-se de processos heurísticos para remover o início e ou o fim da palavra, incluindo afixos derivados. Por exemplo, a palavra “realismo” teria o sufixo “ismo” removido, ficando a forma base “real”.

2.1.2 Indexação de documentos

Dentro da área da recuperação de informação, uma estrutura de dados bastante utilizada para a indexação de documentos é a lista invertida. Essa estrutura de dados mantém um dicionário de termos, sendo que cada termo há uma lista que armazena em quais documentos o termo ocorreu (MANNING; RAGHAVAN; SCHüTZE, 2009). Sua vantagem está em permitir buscas via texto mais rápidas em troca de um maior custo para a adição e atualização de documentos. Tem ampla utilização em motores de busca e sistemas de gerenciamento de banco de dados.

Considerando os documentos na Tabela 2.1, a lista invertida resultante é apresentada na Tabela 2.2. Ao contrário de uma lista tradicional, onde cada entrada é um documento associado a uma lista de termos, na lista invertida, cada entrada é um termo associado à lista de documentos onde ele ocorre.

Tabela 2.1 – Uma coleção de documentos

ID	Texto
1	Futebol é divertido
2	Neymar joga futebol
3	É divertido quando Neymar joga.

Tabela 2.2 – Uma lista invertida

Termo	Documentos
futebol	[1], [2]
é	[1], [3]
divertido	[1], [3]
neymar	[2], [3]
joga	[2], [3]
quando	[3]

2.1.3 Ranqueamento dos resultados

A partir de uma consulta composta por termos, cabe ao sistema devolver ao usuário a lista de páginas *Web* mais relevantes. A ordem das páginas é definida por fatores de ranqueamento. Os fatores podem ser estáticos ou dinâmicos. Os fatores estáticos produzem um ranqueamento, em que a determinação de importância da página independe da consulta. Os fatores dinâmicos produzem um ranqueamento em que a determinação de importância da página pode variar de consulta para consulta.

Ranqueamento Estático

Uma das propostas de ranqueamento estático é denominada *PageRank*(PAGE et al., 1999). O *PageRank* leva em consideração os *links* que "saem" da página, apontando para outras páginas e links que "entram" na página, *links* de outras páginas que apontam para essa página em particular. Essa estrutura de *links* forma um grafo que é utilizado pelo algoritmo *PageRank* para computar a classificação das páginas da *Web*. Desta forma uma página tem uma classificação alta se a soma da classificação dos *links* que apontam para ela é alta.

Para implementar o algoritmo foi utilizado um *web crawler* que criou um repositório de páginas. Em seguida, cada URL é convertida em um valor inteiro único e cada *hiperlink* é armazenado em um banco de dados onde esse valor inteiro é usado como identificador das páginas. Depois disso, a estrutura de *links* é ordenada por identificador. *Links* que apontam para páginas pendentes, isto é, páginas que não apontam para nenhuma outra página, são removidas. Também são atribuídos valores iniciais para a classificação de cada página. Esses valores iniciais não afetam os valores finais de classificação, apenas a sua taxa de convergência, de maneira que a escolha cuidadosa de valores inicias e do número de iterações aumentam o desempenho.

Para avaliar o desempenho do algoritmo, foi implementado um motor de busca que usa apenas os títulos das páginas, de modo que quando é feita uma busca o motor encontra todas as páginas cujos títulos contenham todas as palavras buscadas. Por fim, os resultados

são ordenados pelo algoritmo *PageRank*. Por exemplo, ao buscar pelo termo "university", o algoritmo retornou resultados de maior relevância quando comparado com o motor de busca *Altavista*.

Ranqueamento Dinâmico

Uma das propostas de ranqueamento dinâmico é denominada TF-IDF, que é uma sigla para *term frequency–inverse document frequency*(RAJARAMAN, 2011). O TF-IDF é um número que reflete a importância de uma palavra em um documento dentro de uma coleção de documentos. O valor de TF (*term frequency*) é o número de vezes em que a palavra aparece nos documentos da coleção. Já o valor de IDF (*inverse document frequency)* é o logaritmo, na base 10, do resultado da divisão do número total de documentos da coleção pelo número de documentos em que aparece a palavra. O valor de TF-IDF é calculado multiplicando os valores de TF e IDF.

Dentre os trabalhos acadêmicos que fazem uso do TF-IDF incluem-se (WU et al., 2008) que apresenta um modelo de recuperação probabilístico, (RAMOS, 2003) que aplica o TF-IDF para determinar quais palavras em uma coleção de documentos podem ser mais favoráveis para serem usadas em uma consulta e (ZHANG; YOSHIDA; TANG, 2011) que faz uma comparação entre TF-IDF, LSI (*latent semantic indexing*) e multi-palavra para representação de texto.

Atualmente não se sabe exatamente como o *Google* ranqueia documentos a partir de consultas. No entanto, é provável que a fórmula de cálculo utilize tanto fatores estáticos como dinâmicos, e que TF-IDF e análise dos *hiperlinks* contribua na construção de um valor final de escore.

2.2 Processamento de Documentos

Uma vez recuperados os documentos de interesse, é preciso processá-los para extrair deles as informações pertinentes. Técnicas vistas anteriormente, como preprocessamento, podem ser úteis também nesse momento. Além disso, é importante considerar as diferentes formas de ler um documento. A leitura dos documentos fica a cargo de um processo conhecido como *parsing*.

O *parsing* de documentos é o processamento responsável por analisar um documento e reconhecer as estruturas internas que o compõe. A partir do reconhecimento dessas estruturas, é possível extrair informações específicas que atendam a um propósito. A complexidade do *parsing* depende da natureza dos documentos, que são genericamente classificados em não

estruturados, semi-estruturados e estruturados.

Em documentos não estruturados, as estruturas internas são implícitas, o que exige um processamento mais sofisticado para a sua identificação. Por exemplo, arquivos de texto são compostos por sentenças, sem que haja qualquer espécie de anotação rotulando as sentenças ou os termos que as compõem. Em documentos semi-estruturados, os conteúdos são delimitados por separadores, o que simplifica o processo de organização da informação. No entanto, ainda assim não existem rótulos marcando o conteúdo. Um exemplo são receitas de bolo, em que os ingredientes são separados por quebras de linha. No entanto, ainda é necessário processamento complementar para identificar as subestruturas de cada ingrediente. Já em documentos estruturados existem marcadores que rotulam as informações. Exemplos típicos de documentos estruturados são arquivos XML, onde os nomes dos elementos tem a função de rotulagem. Outro exemplo mais abundante são as próprias páginas HTML, que podem ser vistas como tipos específicos de documentos XML, onde as *tags* tem função tanto de formatação quanto de estruturação do conteúdo.

Para documentos estruturados na forma de XML, existem basicamente duas formas padronizadas de conduzir o *parsing*, denomidadas DOM e SAX. O DOM (*Document Object Model*) é uma interface, independente de linguagem e plataforma, que permite que programas e scripts acessem e atualizem de maneira dinâmica o conteúdo, a estrutura e o estilo de documentos (CONSORTIUM, 2016a). Esse padrão também define a estrutura lógica de documentos, assim como a forma de acesso e manipulação (CONSORTIUM, 2016b). No DOM os documentos são organizados em uma estrutura lógica semelhante a uma árvore onde cada nó é um objeto representando parte do documento, conforme ilustrado na Figura 2.1. Através do DOM, usuários podem criar documentos, navegar em sua estrutura e adicionar, modificar e apagar elementos e conteúdos. O conteúdo de documentos HTML e XML pode ser acessado, modificado, apagado ou adicionado usando o DOM.

Já o SAX (*Simple API for XML*) é uma API baseada em eventos para a interpretação de arquivos XML (PROJECT, 2016). SAX provê um mecanismo para a leitura sequencial de dados de um documento XML, sendo assim uma alternativa ao DOM que não armazena todo o documento em memória. SAX diferencia-se de DOM por operar em cada parte do documento XML sequencialmente enquanto DOM opera no documento em sua totalidade.

Existe diversas implementações de DOM e SAX em linguagens de programação variadas, sendo possível optar pela forma de processamento que se julgar mais conveniente. De

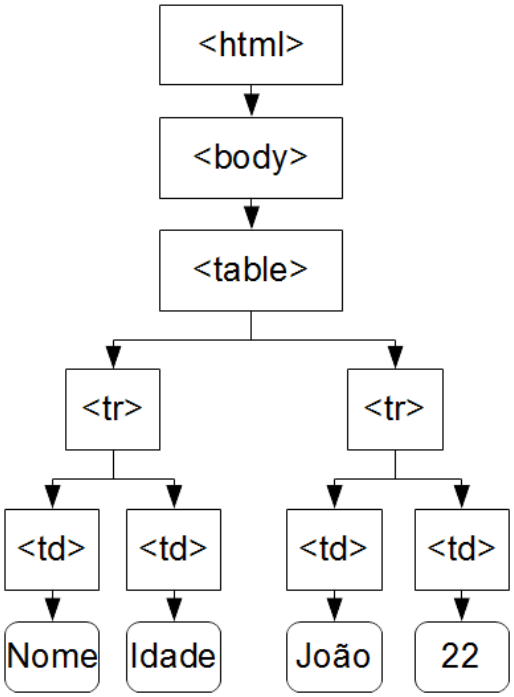

Figura 2.1 – Um exemplo de uma árvore DOM

modo geral, recomenda-se o uso de SAX quando não existe a necessidade em navegar pelos elementos de forma retroativa, e quando o custo em memória para armazenar todo o documento for elevado.

2.3 Resumo do Capítulo

Este capítulo apresentou as teorias que fundamentam o presente trabalho. Entre as técnicas e recursos utilizados pode-se destacar: a recuperação de informação, que provê recursos para a extração de informações relevantes em documentos; o preprocessamento de documentos, onde se utiliza uma lista de *stop words* para filtrar palavras sem relevância; o ranqueamento de resultados, que pode ser dinâmico (*PageRank*) ou estático (TF-IDF); a indexação de documentos, onde se faz uso de listas invertidas e o processamento de documentos, que é utilizado na interpretação de documentos HTML e pode ser feito por DOM ou SAX.

3 TRABALHOS RELACIONADOS

A revisão bibliográfica deste trabalho usou uma abordagem sistemática de pesquisa no site *Google Scholar*, utilizando "trending topics" como termo de pesquisa. Os trabalhos encontrados foram classificados e filtrados. Um resumo de cada um dos artigos considerados relacionados é apresentado abaixo.

Como será visto, grande parte dos trabalhos relacionados é baseado na aplicação de técnicas de mineração de dados. A mineração de dados (ACM; SIGKDD, 2006) é o processo de descoberta de padrões em grandes conjuntos de dados fazendo uso de métodos estatísticos e aprendizado de máquina. O aprendizado de máquina pode ser supervisionado ou não supervisionado. Aprendizado supervisionado (MOHRI; ROSTAMIZADEH; TALWALKAR, 2012) consiste em inferir uma função a partir de um conjunto de dados de treino. Esses dados são um conjunto de pares e cada par é formado por um objeto de entrada e seu valor de saída esperado. Dessa maneira é possível estabelecer uma função que mapeia novas instâncias desconhecidas relativas ao conjunto de dados de treino. Já o aprendizado não supervisionado consiste em inferir uma função de um conjunto de dados não etiquetados. Um exemplo dessa abordagem inclui os algoritmos de agrupamento (DUDA; HART; STORK, 2001).

3.1 Classificação da tipologia de *trending topics*

Zubiaga (ZUBIAGA et al., 2011) apresenta um método para a classificação dos *trending topics* da rede social *Twitter*. A classificação usa quatro tipos: notícias, eventos atuais, *memes* e comemorações.

Para fazer a classificação dos *trending topics* é utilizado o algoritmo de aprendizado supervisionado SVM *(Support Vector Machines)* (STEINWART; CHRISTMANN, 2008). Os atributos usados para a classificação são: nível de *retweets*, proporção entre *retweets* e *tweets*, *hashtags*, comprimento dos *retweets*, exclamações, interrogações, *links*, repetição do *trending topic* no *tweet*, respostas, velocidade de divulgação (*tweets* por segundo), diversidade de usuários contribuintes, diversidade de usuários *retweetados*, diversidade de *hashtags*, diversidade de língua e diversidade de vocabulário.

O experimento usa um conjunto de dados de treino com 600 *trending topics* e 436 no conjunto de dados de teste e os resultados obtidos classificam os *trending topics* com 78,4% de precisão.

3.2 Detecção de bots de spam em redes sociais: uma abordagem com aprendizado de máquina

Wang (WANG, 2010) tem como objetivo detectar *bots* em redes sociais utilizando-se de algoritmos de aprendizado de máquina. Os algoritmos são testados sobre o *Twitter*.

Para formar o *dataset*, é utilizada uma API do *Twitter* e um *Web crawler*. A API do *Twitter* coleta as informações detalhadas do usuário, enquanto o *crawler* coleta os 20 *tweets* mais recentes do usuário.

Os atributos usados para a detecção de spam são: número de amigos, número de seguidores, proporção de seguidores e amigos, número de *tweets* duplicados, número de *links* HTTP e número de respostas/menções.

No processo de avaliação, 500 usuários foram manualmente etiquetados em duas classes: spam e não-spam. Após isso, as predições foram computadas usando validação cruzada com 10 amostras e os quatro algoritmos utilizados foram os seguintes: árvore de decisão, redes neurais, SVM e classificador bayesiano. O algoritmo que obteve melhor resultado é o classificador bayesiano com um *f-measure* de 0,917.

3.3 Além dos Trending Topics: Identificação de Eventos do Mundo Real no Twitter

Becker (BECKER; NAAMAN; GRAVANO, 2011) tem por finalidade identificar de maneira online eventos do mundo real. Os eventos são identificados usando uma técnica de agrupamento que agrupa *tweets* de conteúdo similar.

Primeiramente os *tweets* são agrupados utilizando um algoritmo de agrupamento incremental e a partir dos grupos criou-se quatro tipos de atributos: temporais, sociais, tópicos e centrados no *Twitter*.

O dataset usado nos experimentos consiste em 2.600.000 *tweets* enviados durante o mês de Fevereiro de 2010. Os grupos foram manualmente etiquetados em quatro categorias: eventos do mundo real, atividades do *Twitter*, outros não-eventos e ambíguos. Para a análise dos resultados foi criado um conjunto de treinamento composto por 374 grupos e um conjunto de testes composto por 300 grupos. Após esta etapa utilizou-se o classificador SVM com validação cruzada com 10 amostras para predizer quais grupos representam um evento. Os resultados obtidos mostraram um valor de *f-score* de 0,849 na validação e *0,837* no teste.

3.4 RT para vencer! Predizendo a Propagação de Mensagens no Twitter

Petrović (PETROVIC; OSBORNE; LAVRENKO, 2011) tem como objetivo predizer qual *tweet* será reenviado (retweetado). O conjunto de dados foi formado a partir de 21 milhões de *tweets* usando a API de *streaming* do *Twitter* durante o mês de Outubro do ano de 2010. 90% do conjunto foi usado como treinamento e os restantes 10% foram usados como teste. Para fazer a classificação dos *tweets* o algoritmo selecionado foi o passivo-agressivo de *Crammer* (CRAMMER et al., 2006).

Os atributos usados na classificação dividiram-se em dois tipos: sociais e de *tweet*. Os atributos sociais foram os seguintes: número de seguidores, amigos, status, favoritos, número de vezes que o usuário foi listado, se o usuário é verificado e se o idioma do usuário é inglês. Os atributos de tweet foram os seguintes: número de *hashtags*, menções, URLs, *trending words*, comprimento do *tweet*, se é uma resposta, palavras do *tweet* e novidade.

Além de um modelo global de classificação foi criado também um modelo sensível ao tempo, que consiste em um modelo treinado usando apenas *tweets* criados dentro do período de tempo de uma hora. Esse modelo sensível ao tempo então é composto por 24 modelos, um para cada hora dia.

Os teste ocorreram com o modelo, sensível ao tempo e usando todos os atributos, tendo que, a partir de um par de *tweets*, dentre os quais um foi reenviado e o outro não, escolher qual dos dois foi reenviado. O resultado deste teste mostrou que o modelo obteve uma precisão de 82,7%.

3.5 *Tag Clouds* para Resumo de Resultados em Buscas na *Web*

Kuo (KUO et al., 2007) tem por meta usar uma nuvem de termos (*tag cloud*) para facilitar a navegação e a visualização de resultados em buscas na *Web*. Essa deia deu origem ao sistema chamado *PubCloud*.

A nuvem de termos é gerada a partir de uma busca no sistema *PubMed*, que é um repositório com milhões de publicações biomédicas. Para cada resumo de publicação retornado são removidas as palavras sem significado, pontuações e símbolos.

Em seguida são removidos os sufixos das palavras restantes. Para cada uma dessas palavras é determinada sua frequência pelo seu número de ocorrências, representada pelo tamanho da fonte. As cores usadas nos termos mostram o período de publicação dos resumos. Ter-

mos com vermelho brilhante são os mais recentes e termos com cinza escuro são mais antigos. Apenas os termos que atingem 10% da frequência do termo mais frequente são exibidos.

Para avaliar o trabalho, um grupo de vinte pessoas respondeu a um questionário sobre desenvolvimento de plantas. Metade dos entrevistados usou o sistema *PubCloud* e a outra metade usou apenas a interface padrão do sistema *PubMed*. Os resultados mostraram que o *PubCloud* foi melhor em questões descritivas e pior em identificar relações entre múltiplos conceitos.

3.6 Palavras Tendência em Bibliotecas Digitais para Navegação Baseada em Nuvem de Termos

Molnár (MOLNAR; MORO; BIELIKOVÁ, 2013) busca facilitar a navegação em bibliotecas digitais através do uso de nuvem de termos, levando em consideração os seguintes aspectos: o histórico de navegação do usuário e a posição dos termo.

O conteúdo da nuvem de termos que representa os documentos da biblioteca é criado através de marcações feitas por usuários que descrevem e categorizam esses documentos e também por palavras-chave extraídas dos documentos, como por exemplo, a partir do resumo.

A relevância de cada termo é medida pelo número de vezes em que ele aparece nos documentos. A posição de um termo em uma consulta é usada para classificar a lista de documentos relevantes que contenham pelo menos um dos termos de busca. O conteúdo da nuvem também é adaptado de acordo com o histórico de buscas: quando o usuário modifica uma busca, é feita uma consulta no histórico por buscas semelhantes e os termos mais frequentes nessas buscas são considerados como os mais relevantes. A visualização da nuvem leva em conta o tempo desde a última utilização de cada termo da busca, colorindo-os com uma interpolação de duas cores.

Para avaliar o trabalho foi usado o *Annota*, que é um sistema para criar favoritos e anotações em documentos. O primeiro experimento dividiu os usuários em dois grupos: um com uma interface sem termos do histórico e outro com os termos do histórico. A conclusão é de que o segundo grupo obteve mais facilidade em navegar pelo sistema. O segundo experimento teve a participação de quatro usuários e um conjunto de 4.313 documentos de várias bibliotecas digitais. A conclusão é de que os usuários são mais receptivos às cores dos termos do que ao seu tamanho. Por fim, o terceiro experimento contou com a participação de 17 usuários do sistema *Annota* durante uma semana. Os resultados mostraram que os termos oriundos do histórico de

busca têm o dobro de chance de serem selecionados, sendo eles considerados muito relevantes pelos os usuários.

3.7 *Data Cloud*: Resumindo Buscas por Palavra-Chave em Dados Estruturados

Com o uso frequente de buscas por palavra-chave nas mais diversas aplicações, Koutrika (KOUTRIKA; ZADEH; GARCIA-MOLINA, 2009) propõe criar uma nuvem de termos que sintetize os resultados de buscas feitas em bancos de dados estruturados.

Essa abordagem foi então usada no site *CourseRank* que consiste em um site que mantém um banco de dados sobre as matérias de uma faculdade. Este site permite que os alunos consultem informações sobre as matérias que desejam cursar, podendo inclusive adicionar comentários sobre as mesmas. Os usuários tem a opção de acessar o conteúdo através de duas interfaces: por departamento e por busca por palavra-chave.

A fim de melhorar a experiência do usuário, foi implementado junto à interface do site o *CurseCloud*, que se destaca pelas seguintes características: busca por objetos que se estendem por múltiplas Tabelas, uso de nuvens para condensar resultados de buscas em dados estruturados e uso de nuvens para navegação e refinamento de busca.

A nuvem é formada a partir de uma busca feita no banco de dados e o resultado é uma coleção de entidades de busca. É então atribuído um peso para cada termo nessa coleção baseando-se em três fórmulas: popularidade dos termos, isto é, os termos que mais aparecem nos resultados; relevância dos termos, a partir da similaridade que eles possuem com os termos usados na busca; dependência de busca, que leva em consideração as entidades as quais cada termo pertence.

Para verificar a precisão de cada fórmula, foi realizado um experimento que contou com a participação de três alunos, sendo que cada um formulou 30 consultas diferentes. A partir da análise dos nuvens geradas, concluiu-se que a fórmula de popularidade dos termos obteve precisão abaixo dos 40%, a de relevância obteve 60%, enquanto a dependência de busca ficou com 70%.

3.8 Resumo do Capítulo

Este capítulo apresentou os trabalhos relacionados, que de maneira geral utilizaram-se de técnicas complexas de aprendizado de máquina e mineração de dados exemplificado por

(WANG, 2010) e (PETROVIC; OSBORNE; LAVRENKO, 2011), além de consumir informações a partir de grandes bases de dados, incluindo o *Twitter* como mostra (ZUBIAGA et al., 2011). Isto se contrapõe com a solução proposta no presente trabalho que se vale de técnias simples para o ranqueamento de palavras e que faz uso de base de dados pequenas. Além disso (KUO et al., 2007) e (MOLNAR; MORO; BIELIKOVÁ, 2013) também recorrem ao uso de nuvem de palavras, recurso este que é usado na solução aqui proposta.

4 SEARCH ENGINE TRENDS

Este capítulo apresenta a proposta deste trabalho, denominada *Search Engine Trends*. O objetivo do *SE Trends* é mostrar, para um tema escolhido pelo usuário, palavras que melhor caracterizem esse tema.

Para que esse objetivo seja alcançado, um processo composto de cinco etapas é executado. As etapas que compõem esse processo são as seguintes: carga das páginas, interpretação do HTML, construção do dicionário, cálculo da importância da palavra e formatação dos resultados. O processo e suas etapas são ilustrados na Figura 4.1.

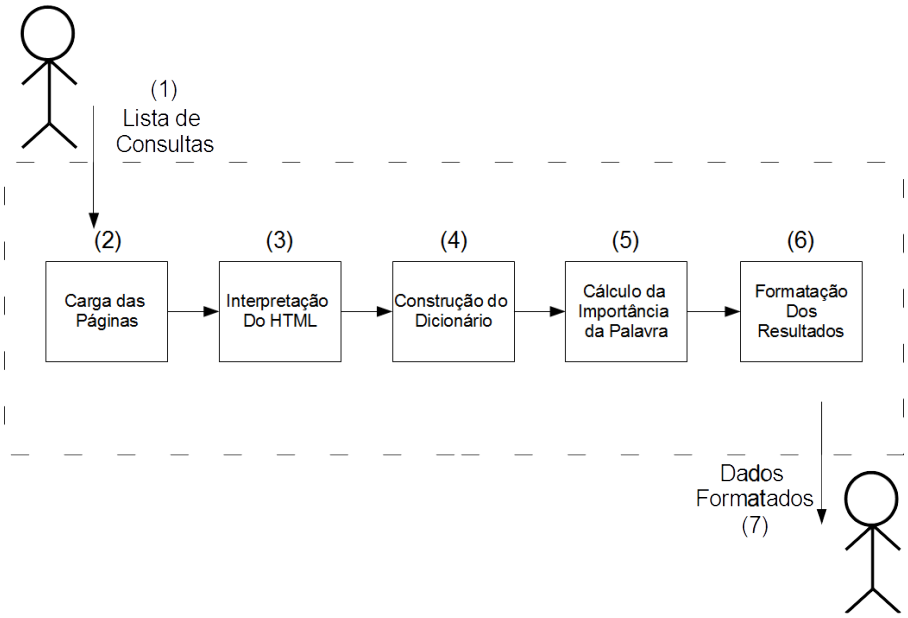

Figura 4.1 – O processo e suas etapas

Em um primeiro momento, cabe ao usuário fornecer uma lista de consultas (1). Cada elemento da lista corresponde a uma consulta individual que será submetida ao motor de busca. Para cada consulta, o parâmetro indispensável são os termos, que representam o tema que se deseja investigar. No entanto, outros parâmetros podem ser fornecidos, como a linguagem das páginas. Na verdade, o suporte à parâmetros é fornecido pelo próprio motor de busca. O SE Trends apenas repassa esses parâmetros para que o motor de busca os utilize.

A etapa de carga das páginas (2) carrega o conteúdo de páginas HTML buscadas por um motor de busca. A etapa de interpretação do HTML (3) interpreta as páginas HTML para que seu conteúdo textual possa ser extraído e processado. A etapa de construção do dicionário (4)

Tabela 4.1 – Parâmetros de consulta suportados pelo motor de busca da *Google*

Parâmetro de Consulta	**Significado**
q	Termo da consulta a ser pesquisado
key	Chave da API
cx	Identificador do motor de busca
alt	Formato da resposta
lr	Idioma dos resultados retornados

constrói um dicionário a partir do texto das páginas previamente buscadas. A etapa de cálculo da importância da palavra (5) atribui um valor para cada palavra que compõe o texto, utilizando uma medida estatística. Finalmente, as palavras consideradas mais importantes são formatadas e exibidas ao usuário (6) (7).

As seções seguintes descrevem em detalhe cada uma dessas etapas que compõem o processo.

4.1 Carga das Páginas

O processo de carga das páginas consiste em carregar o conteúdo HTML de páginas *Web* para que este possa ser processado. A carga ocorre a partir de consultas executadas sobre o motor de busca da *Google*. Essa escolha deve-se ao fato de que esse é o motor de busca mais usado nos dias de hoje (APPLICATIONS.COM, 2016).

Nesse motor de busca, uma consulta é composta por termos e por parâmetros opcionais. A Tabela 4.1 apresenta parâmetros que podem ser especificados.

Para exemplificar as atividades que estão por trás do processo de carga, considere a Figura 4.2, que apresenta os 5 primeiros resultados de uma consulta na *Google*. No exemplo, um usuário escreveu a consulta em um navegador e visualiza os links de resposta. Se precisar, o usuário tem a opção de seguir os links para ter acesso ao conteúdo dos documentos propriamente ditos. Dentro do SE Trends, o desafio passa a ser encontrar uma maneira de submeter essa consulta e consumir os resultados retornados programaticamente, com pouca intervenção manual.

O problema pode ser dividido em duas partes. A primeira envolve a submissão de uma consulta e obtenção dos resultados. A segunda envolve o processamento dos resultados.

Três alternativas foram estudadas para realizar a submissão da consulta e obtenção dos resultados: a API *Google Web Search*, a API *Google Custom Search* e um *web crawler*.

A API *Google Web Search* permitia fazer buscas no motor de busca da *Google*. Porém,

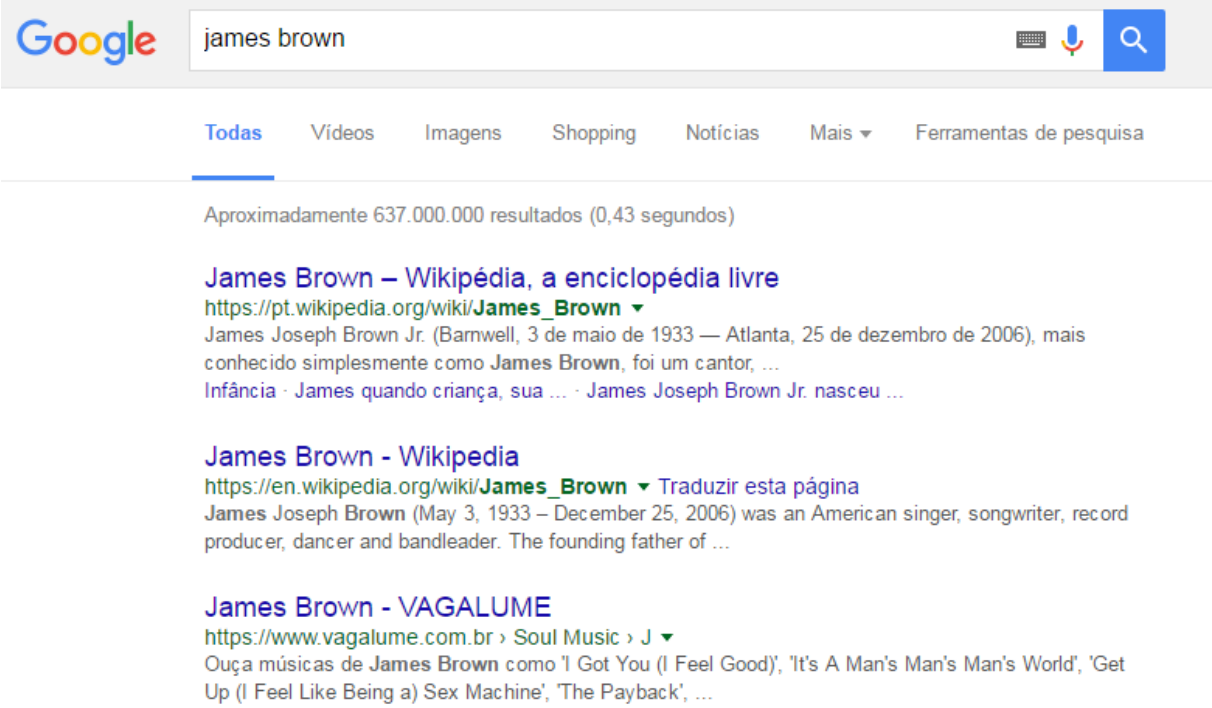

Figura 4.2 – Uma consulta realizada por um navegador web

a API foi descontinuada em 1 de novembro de 2010 e encerrou sua operação em 29 de setembro de 2014.

A API *Google Custom Search* permite fazer uma busca no *Google* restrita a um domínio específico. Entretanto, pode-se evitar esta restrição através do painel de controle da API onde pode-se remover a limitação de domínio. Para realizar uma busca, é utilizada uma requisição HTTP GET para o URI correspondente, que retorna um arquivo JSON ou *Atom*, de acordo com a opção desejada. O trecho de código 1 mostra um exemplo de consulta na API da linguagem *Python*. No exemplo, o termo consultado foi "futebol". A desvantagem deste método é o limite de 100 requisições por dia.

Outra forma de se fazer essa consulta é através de um *web crawler*, usando apenas as ferramentas da própria linguagem. Isso envolveria processar a página de resposta através de um parser HTML para reconhecer os *links* que compõe a resposta.

Dentre os métodos estudados foi então escolhida a API do *Google Custom Search*. O motivo deve-se à simplicidade no acesso. O limite de acessos diários não foi um fator impactante para a realização do trabalho.

A carga das páginas é então realizada buscando-se os termos fornecidos pelo usuário utilizando o *Google Custom Search*. Para cada consulta que o usuário submete, a API retorna

um arquivo textitAtom contendo as URLs das páginas de resposta. O arquivo *Atom* é então processado por um parser XML, para que as URLs sejam extraídas. É importante salientar que o arquivo *Atom* retorna as 10 URLs consideradas mais relevantes (as top-10 páginas). São essas 10 URLs que serão processadas. O acesso às URLs que ficam além desse limite está fora do escopo deste trabalho. Uma sugestão de aprimoramento é o uso de um parâmetro que indique quantas URLs devem ser processadas para cada consulta submetida, para que a análise não fique restrita a um valor constante de páginas.

Para cada URL presente nas top-10 é carregado o seu conteúdo HTML correspondente através de uma requisição HTTP GET. Este conteúdo HTML é posteriormente processado em um interpretador HTML, como será descrito a seguir.

Algoritmo 1 Uma consulta pela palavra "futebol" na API Google Custom Search

```
chave_api = (...)
id_buscador = (...)
atom = urlopen('https://www.googleapis.com/customsearch/v1?
   key=chave_api&cx=id_buscador&q=Futebol&alt=atom&lr=
   lang_pt')
resultado_consulta = atom.read()
```

4.2 Parsing do HTML

A etapa anterior consegue recuperar, para cada página de resposta, o seu conteúdo em formato textual. Assim, é necessário que haja um processamento capaz de ler as páginas e identificar as partes que sejam relevantes. No contexto deste trabalho, são considerados relevantes os conteúdos textuais da página, tipicamente escritos na forma de parágrafos. Ou seja, textos referentes a linguagens de *script* e textos encapsulados em Tabelas são considerados irrelevantes e não devem ser processados.

Conforme discutido no Capítulo 2, as formas padronizadas de processar documentos HTML são por meio das APIs de *parsing* SAX e DOM. Neste trabalho optou-se por utilizar DOM, uma vez que os documentos HTML não consomem um valor proibitivo de memória.

Duas APIs DOM foram testadas: *DOMDocument*, da linguagem PHP, e *Beautiful Soup*, da linguagem *Python*. A API para PHP mostrou-se limitada para o processamento de páginas que continham códigos em *javascript*. Dessa forma, optou-se pelo uso da API para *Python*. Observa-se que essa decisão norteou todo o restante do trabalho, pois ela definiu a linguagem

de programação que seria utilizada em todos os processos.

O *Parser* DOM cria uma representação do documento em memória na forma de uma árvore onde os nós são as tags HTML e seus respectivos conteúdos. A partir do nó raiz, os demais nós são acessados recursivamente, de modo a possibilitar o acesso a todas as *tags* do documento. O objetivo é extrair o conteúdo textual de cada nó, desde que ele seja considerado relevante.

Para a interpretação das páginas HTML foram estabelecidas algumas regras que definem quais partes do documento teriam o seu conteúdo efetivamente processado. Para a aplicação das regras, algumas *tags* HTML são classificadas como *permitidas* e textitproibidas. Ao encontrar uma *tag* que atenda essa classificação, ações específicas são realizadas, conforme descrito a seguir.

- **tags permitidas:** Quando uma *tag* permitida for encontrada, o seu conteúdo textual é extraído. Exemplos de *tags* permitidas são aquelas que costumam encapsular parágrafos, como <*body*> e <*p*>.

- **tags proibidas:** Quando uma *tag* proibida for encontrada, tanto seu conteúdo textual quando suas *tags* descendentes são ignorados, pois elas levam a partes do documento que não possuem parágrafos de texto. Exemplos de *tags* proibidas são <*script*> e <*option*>.

A Figura 4.3 mostra um exemplo de HTML onde é destacado em azul as *tags permitidas* e em vermelho as *tags proibidas*. Já a Figura 4.4 mostra em destaque o conteúdo a ser extraído depois de aplicadas as regras de processamento. O conteúdo extraído será utilizado posteriormente para a construção do dicionário de palavras.

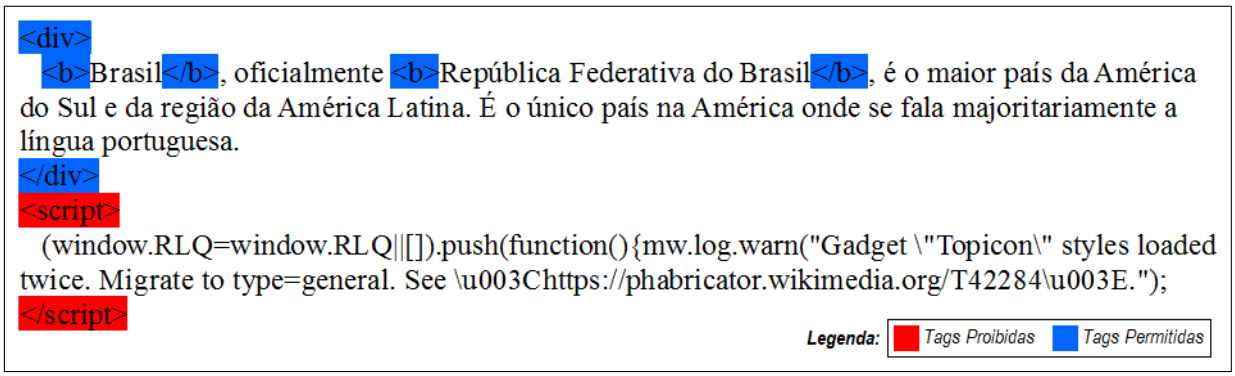

Figura 4.3 – Código HTML com as tags *permitidas* e *proibidas* realçadas

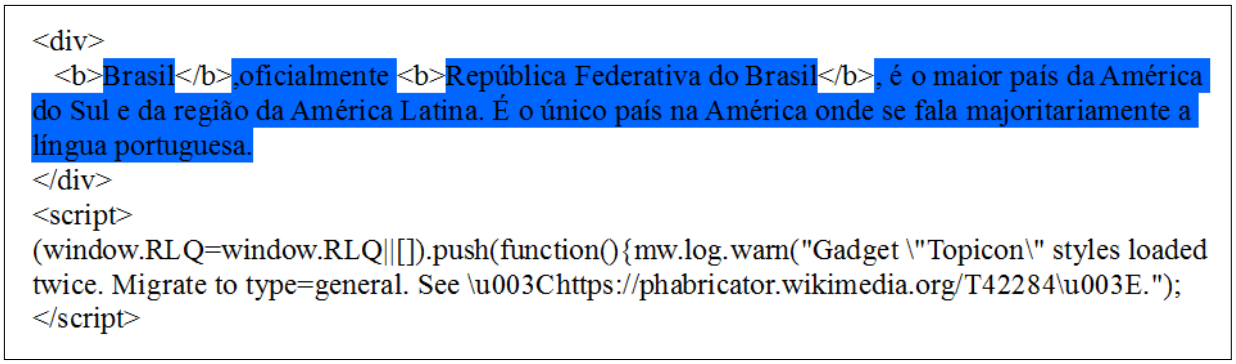

Figura 4.4 – O texto extraído após processamento com destaque em azul

4.3 Construção do Dicionário

O texto extraído das páginas HTML é utilizado na construção de um dicionário de palavras. Esse dicionário é composto por palavras consideradas relevantes para o processo de ranqueamento. Assim, para garantir que apenas palavras relevantes sejam processadas, o texto é preprocessado através de uma técnica de remoção de *stop words*.

Para a aplicação da técnica, é mantida uma lista de *stop words*, composta por palavras comuns que não apresentam um significado relevante, tais como artigos, preposições e conjunções. Palavras contidas nessa lista não são processadas. Além disso, caracteres isolados (ex. 'x', 'y') ou palavras compostas por caracteres especiais (ex. '*&$') também são descartadas de tal maneira que apenas palavras com dois ou mais caracteres são processadas.

A partir do texto preprocessado, é realizada a construção do dicionário. O pseudocódigo 2 mostra como o dicionário é construído. Inicialmente, um dicionário auxiliar é usado. Esse dicionário guarda o número de vezes que cada palavra ocorreu em cada documento. A partir dessa contagem, inicia-se a segunda parte do algoritmo, que constrói o dicionário propriamente dito. Para cada palavra, o dicionário armazena o número de vezes que a palavra ocorreu e o número de documentos em que ela ocorreu.

A Tabela 4.2 mostra um dicionário construído a partir de um conjunto qualquer de documentos. A primeira coluna da Tabela 4.2, denominada *Palavra*, representa uma entrada de palavra no dicionário. A segunda coluna, denominada *Frequência da Palavra*, é o número de vezes em que essa palavra apareceu no total de documentos. A terceira coluna, denominada *Frequência de Documentos*, é o número de documentos em que a palavra esteve presente.

Depois de construído, o dicionário é usado para a formatação do ranking de palavras, conforme será discutido na próxima seção. Cabe ressaltar que o dicionário é construído a partir

Algoritmo 2 Pseudocódigo de criação do dicionário

```
para cada documento faça
    para cada palavra faça
        dic_local[documento][palavra] += 1
    fim para
fim para
para cada documento faça
    para cada palavra faça
        dic[palavra].cont += dic_local[documento][palavra]
        se dic_local[documento][palavra] > 0 então
            dic[palavra].contDoc += 1
        fim se
    fim para
fim para
```

Tabela 4.2 – Dicionário criado a partir de uma coleção qualquer de documentos

Palavra	Frequência da Palavra	Frequência de Documentos
música	53	9
disco	18	6
brasil	15	4
cantor	22	3
mpb	10	1

da lista completa de consultas submetidas e das páginas retornadas para cada consulta. Assim, se cada consulta retornar 10 URLs, e cinco consultas forem utilizadas, o dicionário será construído a partir da análise de 50 páginas de resposta.

4.4 Cálculo da Importância da Palavra

Para determinar a relevância de cada palavra é necessário que seja atribuído um valor de escore a cada uma delas. Para isso, foram escolhidos o TF-IDF e a frequência simples como medida de importância para a palavra.

O valor de cada palavra é atribuído após a construção do dicionário. Caso o usuário opte por frequência simples, o escore associado é o número de vezes em que a palavra ocorreu em todos os documentos. Caso o usuário opte por TF-IDF, o escore associado é o produto da frequência da palavra pelo IDF.

O valor do IDF é definido pelo logaritmo na base 10 do quociente entre número total de documentos processados na consulta pelo número de documentos em que a palavra ocorreu. A fórmula (4.1) mostra o cálculo do TF-IDF para uma palavra p, onde f_p é a frequência da palavra, D é o número total de documentos e d_p é o número de documentos onde a palavra p

Tabela 4.3 – Valor do TF-IDF das palavras da Tabela 4.2

Palavra	**TF-IDF**
música	2,42
disco	3,99
brasil	5,96
cantor	11,50
mpb	10,00

ocorre.

$$\text{TF-IDF}(p) = f_p \times \log \frac{D}{d_p} \tag{4.1}$$

A Tabela 4.3 mostra o valor do TF-IDF baseado em palavras contidas em uma hipotética coleção composta por dez documentos.

Ao contrário do que é comumente feito no cálculo do TF-IDF, a frequência utilizada é a total de todos os documentos nos quais a palavra ocorre, ao invés de se utilizar apenas a frequência de um único documento. Essa mudança é necessária porque não se pretende descobrir a importância de um termo para um documento específico, mas sim, a importância do termo para todo o conjunto de documentos recuperados.

4.5 Formatação dos Resultados

Depois de calculada a importância da palavra, o resultado é exibido ao usuário por meio de uma nuvem de palavras. A nuvem de palavras é gerada usando a lista de palavras criada, como descrito na seção 4.4. Cada palavra na nuvem recebe um destaque de acordo com o seu escore associado. Quanto maior o escore, maior a evidência dessa palavra na nuvem.

Para que o usuário possa interagir com o SE Trends, foi criada uma interface gráfica para a ferramenta, utilizando o pacote gráfico *TkInter* (FOUNDATION, 2016) presente na linguagem *Python*. Com esta interface, o usuário pode entrar com a lista de consultas do tema que deseje pesquisar e visualizar os resultados obtidos.

A tela da ferramenta é dividida horizontalmente em duas partes, como mostra a Figura 4.5. A parte superior apresenta ao usuário os parâmetros para a composição das consultas. A metade inferior apresenta a nuvem de palavras gerada a partir das consultas fornecidas pelo usuário.

A parte superior é composto por uma lista de consultas, onde cada linha traz uma con-

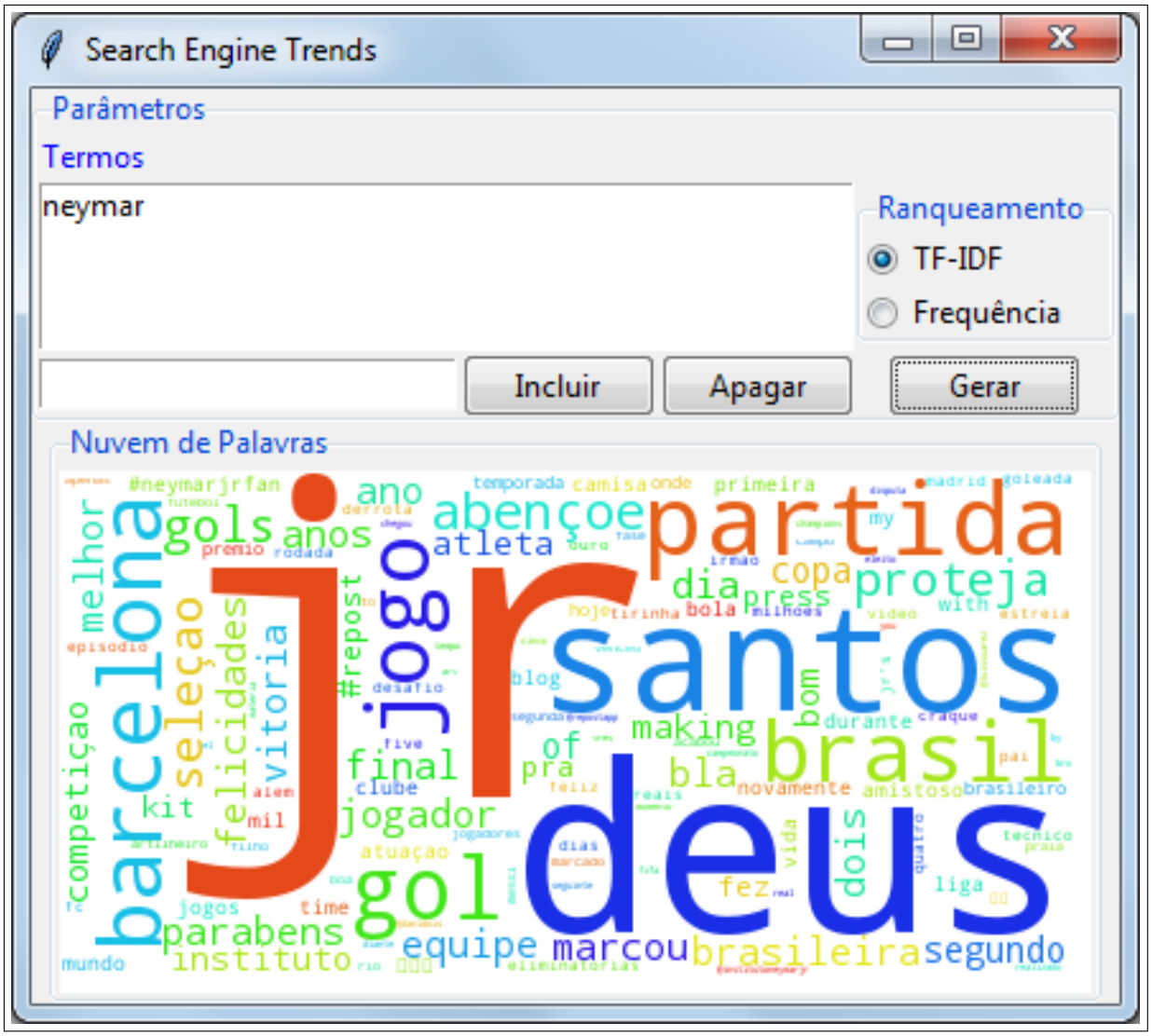

Figura 4.5 – A tela do aplicativo com suas funcionalidades

sulta composta por termos; uma caixa de texto para a digitação dos termos de uma consulta a ser inserida na lista; botão para inserir a consulta na lista; botão para remover da lista a consulta selecionada; dois botões radiais para a escolha do tipo de ranqueamento; e por fim o botão para gerar a nuvem de palavras.

Para exemplificar, a Figura 4.5 mostra o resultado gerado para uma única consulta composta pelo termo 'Neymar'. Na nuvem, as palavras mais evidentes são exibidas em uma fonte maior. Já as cores são escolhidas aleatoriamente no momento em que a nuvem de palavras é criada.

4.6 Resumo do Capítulo

Este capítulo apresentou a estrutura da solução proposta na qual ocorrem cinco etapas: a carga de páginas HTML, onde as páginas são carregadas com auxílio da API Google Custom Search; a interpretação do HTML, onde o contéudo das páginas é interpretado usando um método DOM; a construção do dicionário, onde as palavras são armazenadas em um dicionário juntamente com a sua frequencia de documentos; o cálculo da importância da palavra, onde é atribuído um valor a cada palavra que pode ser baseado no seu TF-IDF ou na sua frequencia e a formatação dos resultados, que apresenta ao usuário via uma interface gráfica uma nuvem de palavras com os resultados obtidos.

5 AVALIAÇÃO EXPERIMENTAL

Este capítulo apresenta os resultados atingidos com o uso do SE Trends. Três cenários distintos são avaliados. No primeiro, pretende-se verificar as diferenças entre usar TF-IDF e somente frequência como fator de ranqueamento. No segundo, pretende-se verificar as diferenças entre usar consultas com termos isolados e com termos combinados. No terceiro, pretende-se verificar o tipo de retorno produzido quando o assunto pesquisado são adjetivos.

Os resultados são analisados através de uma comparação o serviço *Google Trends*. Esse serviço, assim como o SE Trends, exibe listas de palavras relacionadas à temas indicados pelo usuário, sendo as palavras apresentadas por ordem de importância. Isso permite uma comparação direta, uma vez que o tipo de retorno é idêntico.

5.1 Cenário 1: Uso de Somente Frequência e TF-IDF

O objetivo deste experimento é analisar as diferenças quando se usa somente frequência ou TF-IDF para ranquear as palavras. A lista de consultas submetida foi composta por apenas uma consulta. O termo "Dilma" foi usado nessa consulta. A escolha deve-se pela popularidade do termo, associado a uma importante personagem da política brasileira.

A Tabela 5.1 exibe os termos mais relevantes considerando as duas variações do SE Trends e o *Google Trends*. Os números entre parênteses indicam o escore utilizado para o ranqueamento. Por exemplo, o termo "Rouseff" apareceu 79 vezes em todos os documentos recuperados a partir da busca.

Tabela 5.1 – As cinco palavras mais bem ranqueadas por TF-IDF, Frequência e *Google Trends*

SE Trends: TF-IDF	**SE Trends: Frequência**	**Google Trends: Tópicos Relacionados**
outubro (23.51)	rousseff (79)	Impeachment - Tópico
araujo (21.94)	presidente (63)	Mandioca - Planta
março (20.99)	governo (60)	Eduardo Cunha - Ex-Presidente da Câmara
dilmabr (18.13)	brasil (41)	Lepo Lepo - Tópico
durante (12.40)	segundo (40)	Sérgio Moro - Juiz Federal

Uma primeira análise mostra que o Google Trends é capaz de identificar tópicos compostos por mais do que uma palavra, e classifica os tópicos como entidades nomeadas. Já a versão implementada do SE Trends só permite a indexação de termos simples. A engenharia por trás do *Google Trends* é desconhecida. É possível que seja usado um processo semi-automático

que conte com intervenção humana para tarefas complexas, como por exemplo, para a definição dos tópicos e sua posterior categorização. De qualquer forma, a extensão do SE Trends para permitir tópicos compostos é um tema interessante para trabalhos futuros.

Como pode-se ver, os termos de maior importância mudam consideravelmente de uma estratégia para a outra. O serviço da *Google* recuperou um evento (*Impeachment*), duas pessoas ("Eduardo Cunha" e "Sérgio Moro"), um objeto ("mandioca") e uma música ("Lepo Lepo") relacionados à consulta. Em todos os casos, os tópicos retornados realmente possuem uma alto grau de relevância com o tema consultado. Curiosamente, o objeto e a música são referências pejorativas.

O SE Trends com TF-IDF retornou duas referências temporais ("outubro" e "março"), uma pessoa ("araujo"), um domínio *Web* ("dilmabr") e uma proposição ("durante"). As referências temporais são relativas à épocas em que ocorreram eventos importantes [1]. A pessoa retornada é o ex-marido de Dilma. Ao analisar o conteúdo das páginas que levaram até esse resultado, percebeu-se que o termo "Araujo" ocorreu muitas vezes em um único documento. Assim, ele foi considerado um termo raro, mas com frequência elevada, o que justifica a importância atribuída. Vale a pena destacar que, enquanto o *Google* recuperou pessoas com relacionamento de cunho político, o SE Trends retornou um relacionamento afetivo. O domínio *Web* refere-se à paginas que retratam a vida de Dilma. Já a preposição retornada pode ser considerada uma anomalia, que poderia ser eliminada com o enriquecimento da lista de *stop words*.

O SE Trends com frequência retornou termos que costumam acompanhar a palavra "Dilma". De certa forma, se vistos juntos, os termos relacionados oferecem uma descrição sumarizada a respeito de quem é "Dilma", informando seu sobrenome, a função desempenhada("presidente" e "governo") e o país onde essa função foi desempenhada ("Brasil"). Mais uma vez uma preposição apareceu como tópico ("segundo"). Como a lista de *stop words* não é exaustiva, ocorrências como essa não foram eliminadas por completo.

As Figuras 5.1 e 5.2 apresentam os tópicos retornados pelo SE Trends na forma de nuvem de palavras. Esse tipo de exibição consegue descrever de forma visual como os termos importantes estão relacionados, usando para isso um espaço relativamente curto, o que facilita a visualização. Por exemplo, a nuvem da Figura 5.2 mostra que os termos "lula" e "impeachment" aparecem com um grau semelhante de importância. Seria mais difícil fazer essa mesma constatação se os diversos tópicos encontrados fossem exibidos na forma de uma lista ordenada.

[1] em outubro, ocorreram as duas eleições de Dilma à presidêcnia do Brasil. Em março, ocorreram manifestações públicas contrárias ao governo

Figura 5.1 – Consulta pelo termo “dilma”, classificada por TF-IDF

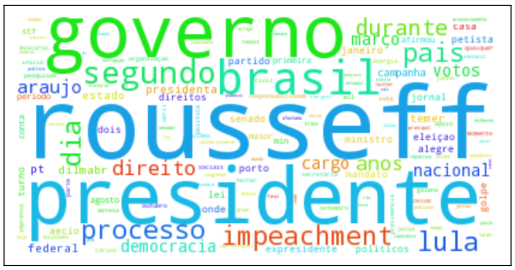

Figura 5.2 – Consulta pelo termo “dilma”, classificada por frequência

5.2 Cenário 2: Uso de Termos Isolados e Termos Combinados

Conforme mencionado no Capítulo 4, o SE Trends permite que seja submetida mais de uma consulta de uma só vez para a geração dos resultados. Esse recurso permite que se compare as diferenças entre usar uma única consulta que contem termos combinados e usar consultas distintas onde cada termo aparece de forma isolada.

Para avaliação, o teste compreende os termos “gremio” e “inter”. A escolha deve-se pela popularidade dos termos, associado a importantes times de futebol da região sul do país. A Figura 5.3 mostra o resultado gerado quando é realizada uma consulta para cada termo. Já a Figura 5.4 mostra o resultado gerado quando a consulta compreende os dois termos (“gremio inter”). Em ambos os casos foi utilizado o TF-IDF para fins de ranqueamento.

Figura 5.3 – Consulta pelo frase “gremio inter’.

Figura 5.4 – Consulta pelas palavras “gremio” e “inter”, combinadas.

A Tabela 5.2 mostra um comparativo entre as 10 primeiras respostas retornadas pelo SE Trends e pelo *Google Trends*. A coluna da direita refere-se ao SE Trends onde os termos “Gremio” e “Inter” são usados em consultas distintas. A coluna do meio refere-se ao SE Trends onde os termos “Gremio” e “Inter” aparecem na mesma consulta. A coluna da direita refere-se ao *Google Trends* onde os dois termos aparecem juntos.

No *Google Trends*, muitas entradas são associadas ao mesmo tópico (“campeonato brasileiro”), diferenciados apenas pelo ano de ocorrência. Também apareceram jogadores e clubes

de futebol que possuem associação à pelo menos um dos termos da consulta. No entanto, o serviço não percebe de maneira inequívoca que “Inter” é uma referência a um time da região sul do Brasil. Isso levou a exibição de um clube europeu (“NK Inter Zapresic”) e nenhuma menção ao clube realmente vinculado à consulta. Essa inferência poderia ser feita se os termos “inter” e “gremio” fossem vinculados para definir um contexto.

Tabela 5.2 – Comparativo entre as consultas e o *Google Trends*

SE Trends: (“gremio”, “inter”)	**SE Trends: (“gremio inter”)**	**Google Trends: (“gremio inter”)**
campeao	internacional	Campeonato Brasileiro de Futebol 2015 - Série A
clube	grenal	Campeonato Brasileiro de Futebol 2014 - Série A
estadio	partida	Campeonato Brasileiro de Futebol 2012 - Série A
internazionale	grenais	Campeonato Brasileiro de Futebol 2013 - Série A
gaucho	maior	Luan Guilherme de Jesus Vieira
final	arbitro	NK Inter Zapresic
ano	gols	Wanderson Ferreira de Oliveira
segundo	historia	Copa Libertadores da América 2015
equipe	classico	Giuliano Victor de Paula
temporada	jogo	NK Zagreb

No SE Trends usado com termos separados foi observado um fenômeno semelhante. O tópico “Internazionale” refere-se a um clube italiano, cuja afinidade com o clube gaúcho resume-se à grafia do nome. Mais uma vez, o problema está associado à falta de vinculação entre os termos, uma vez que eles foram usados em consultas diferentes. Já com o SE Trends com termos compostos, todas respostas estão associadas a pelo menos um dos termos usados na consulta. Além disso, aparecem respostas são de fato relevantes quando os dois termos são colocados em um contexto, como por exemplo, “clássico”, “grenal” e “grenais”.

Nesse exemplo fica clara uma diferença entre as abordagens avaliadas. Enquanto a Google Trends retorna entidades, a ferramenta SE Trends retorna palavras, independente de sua categoricação. Isso acarreta em respostas que não despertam interesse prático, como adjetivos (ex. “maior”) e palavras despidas de contexto (ex. “ano”). Essas ocorrências não são vistas como uma limitação, mas uma forma alternativa de entender um assunto. Afinal, tratam-se de palavras que apareceram em documentos relevantes associados ao assunto. Uma possibilidade de aprimoramento é analisar como esses termos são usados nos documentos para reconstruir a resposta usando entradas mais informativas. Além disso, outra ideia a se investigar é permitir que a lista de *stop words* seja retroalimentada pelos usuários para que determinados termos passem a ser ignorados.

5.3 Cenário 3: Uso de Termos Adjetivos

O objetivo desse experimento é avaliar os resultados gerados quando as consultas são compostas por termos adjetivos. Para fins de avaliação, foram submetidas duas consultas separadas. Na primeira consulta, o termo usado foi “bela”. Na segunda, o termo usado foi “organizada”. A intenção é verificar se as ferramentas avaliadas conseguem associar os adjetivos a algum substantivo, como locais ou pessoas. Em ambas as consultas foi utilizada a classificação por TF-IDF.

A Figura 5.5 mostra a nuvem de palavras resultante das consultas. A Tabela 5.3 mostra as cinco palavras com melhor classificação. Para essa consulta específica, o *Google Trends* não trouxe nenhuma resposta.

Figura 5.5 – Consultas pelas palavras “bela” e “organizada” combinadas em uma nuvem de palavras.

Os resultados mostram que os termos relevantes não possuem uma associação direta aos substantivos consultados. Além disso, os termos não estão concentrados em um domínio específico. Para compreender como se chegou até esse ranqueamento, foram analisadas as páginas retornadas pelo motor de busca. Foi possível perceber que poucas páginas contribuíram de maneira significativa para que alguns termos se sobressaíssem aos demais. Por exemplo, a página (http://vidaorganizada.com/) concentrou 68% de todas as ocorrências do termo mais relevante (“ano”). Ou seja, essa página foi a principal responsável pelo ranqueamento obtido

para esse termo.

O favorecimento dos termos que aparecem muitas vezes em poucos documentos é uma consequência direta do uso do TF-IDF. Para evitar que alguns documentos influenciem demais no cálculo do escore, é necessária a criação de um novo fator de ranqueamento que favoreça os termos que apresentem uma maior regularidade de frequência.

Tabela 5.3 – As cinco palavras melhores classificadas a partir das consultas “bela” e “organizada”.

Palavra	**TF-IDF**
ano	41.23
festa	31.90
blog	31.30
camisa	29.00
vida	28.78

5.4 Resumo do Capítulo

Este capítulo apresentou os experimentos para avaliar a ferramenta implementada pelo presente trabalho. Os experimentos usaram como base de comparação o *Google Trends* e pode-se concluir que a ferramenta obtém êxito em identificar palavras relevantes dado um determinado tema ao passo que o *Google Trends* identifica entidades compostas, ela também consegue fazer associações entre dois termos quando estes são buscados em conjunto, diferentemente do *Google Trends* que não é capaz de fazer tais associações. Entretanto a ferramenta apresenta problemas no seu ranqueamento por TF-IDF em ocasiões em que é dado um valor muito alto a palavras que aparecem de maneira frequente em poucas páginas em detrimento de palavras que aparecem em muitas páginas.

6 CONCLUSÃO

Por meio deste trabalho desenvolveu-se a ferramenta Search Engine Trends (SE Trends), que possibilita a sumarização de informações na *Web*, acerca de um tema no qual o usuário tenha interesse. Para alcançar esse objetivo, o SE Trends utilizou técnicas de recuperação de informação, processamento e indexação de documentos, ranqueamento de palavras e formatação da resposta através de nuvens de palavras.

Os experimentos realizados apresentaram resultados bastante relevantes, trazendo palavras de grande significância em relação ao termo buscado, como constatado na seção 5.1. A capacidade de associar os termos buscados quando os mesmos são usados em uma única frase pode ser vista na seção 5.2, situação na qual o *Google Trends* não obteve o mesmo êxito.

O ranqueamento por TF-IDF apresentou alguns problemas quando palavras com uma alta frequência eram oriundas de poucas ou apenas uma página, especialmente quando o termo buscado não representava uma entidade, como mostrado na seção 5.3. Deste modo, torna-se necessária a criação de uma nova forma de ranqueamento que possa tratar deste tipo de caso.

Outro aprimoramento que pode ser feito diz respeito à atribuição de uma maior semântica às palavras retornadas, de modo a ajudar o usuário a identificar o assunto que existe por trás de cada palavra. Essa tarefa pode ser realizada através da identificação de entidades formadas por palavras compostas, ou através de mecanismos que rotulem as palavras retornadas com o assunto a que elas se referem. A resposta gerada também pode ser enriquecida através de um mecanismo de *feedback*, que permita ao usuário reportar palavras que sejam *stop words*, para que elas não apareçam novamente em consultas futuras.

Além disso, a ferramenta ainda carece de alguns ajustes relativamente simples que poderiam torná-la mais completa. Como mencionado no texto, a entrada é um conjunto de consultas compostas por termos e por parâmetros de consulta. Esse parâmetros podem ser usados para um melhor direcionamento da busca, como por exemplo, limitando a busca em sites de notícia e/ou páginas publicadas em períodos específicos de tempo. No entanto, a API usada (*Google Custom Search*) não proporciona os mesmos recursos que existem quando se usa o motor de busca diretamente pelo navegador *web*. Uma alternativa simples para contornar esse problema é a substituição do módulo de carga das páginas, para que um *Web crawler* se encarregasse de realizar a busca.

Outro aspecto que vale a pena mencionar é que o desempenho da ferramenta é muito

dependente da qualidade da rede disponível, pois isso afeta o tempo necessário para fazer as cargas das páginas *Web*. Tipicamente, o tempo gasto em média para processar uma consulta simples a um único termo é de 30 segundos, variando conforme a velocidade da rede.

Esse custo pode inviabilizar a ferramenta como um mecanismo de busca em tempo real. No entanto, ela ainda pode atrair o interesse quando a resposta gerada é mais importante do que a capacidade de gerar um retorno imediato. Por exemplo, empresas podem usar esse tipo de serviço para a elaboração de campanhas de marketing ou campanhas publicitárias. Como o SE Trends e o *Google Trends* geram respostas diferentes, é possível que essas informações complementares sejam usadas em conjunto, dando maior embasamento na determinação dos tópicos de maior relevância.

REFERÊNCIAS

ACM; SIGKDD. **Data Mining Curriculum**. Acessado em Maio/2016, http://www.kdd.org/curriculum/index.html.

APPLICATIONS.COM, N. **Search engine market share**. Acessado em Dezembro/2016, https://www.netmarketshare.com/search-engine-market-share.aspx?qprid=4&qpcustomd=0.

BEYOND TRENDING TOPICS: REAL-WORLD EVENT IDENTIFICATION ON TWITTER. **Anais...** [S.l.: s.n.], 2011. p.438–441.

CONSORTIUM, W. W. W. **WC3 Documet Object Model**. Acessado em Novembro/2016, https://www.w3.org/DOM/.

CONSORTIUM, W. W. W. **What is the Document Object Model**. Acessado em Novembro/2016, https://www.w3.org/TR/WD-DOM/introduction.html.

CRAMMER, K. et al. Online passive-aggressive algorithms. **The Journal of Machine Learning Research**, [S.l.], v.7, p.551–585, 2006.

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern Classification**. 2nd.ed. [S.l.]: Wiley, 2001.

FOOTE, J. An overview of audio information retrieval. **Multimedia systems**, [S.l.], v.7, n.1, p.2–10, 1999.

FOUNDATION, P. S. **25.1. tkinter - Python interface to Tcl/Tk - Python 3.5.2 documentation**. Acessado em Dezembro/2016, https://docs.python.org/3.5/library/tkinter.html.

FRAKES, W. B. **Information Retrieval**: data structures & algorithms. [S.l.]: Prentice Hall, 1992.

GOODRUM, A. A. Image information retrieval: an overview of current research. **Informing Science**, [S.l.], v.3, n.2, p.63–66, 2000.

JANSEN, B. J.; RIEH, S. Y. The seventeen theoretical constructs of information searching and information retrieval. **Journal of the American Society for Information Science and Technology**, [S.l.], v.61, n.8, p.1517–1534, 2010.

KOUTRIKA, G.; ZADEH, Z. M.; GARCIA-MOLINA, H. Data clouds: summarizing keyword search results over structured data. In: INTERNATIONAL CONFERENCE ON EXTENDING DATABASE TECHNOLOGY: ADVANCES IN DATABASE TECHNOLOGY, 12. **Proceedings...** [S.l.: s.n.], 2009. p.391–402.

KUO, B. Y. et al. Tag clouds for summarizing web search results. In: WORLD WIDE WEB, 16. **Proceedings...** [S.l.: s.n.], 2007. p.1203–1204.

LAB, P. S. **The Internet in Real-Time**. Acessado em Maio/2016, http://pennystocks.la/internet-in-real-time/.

LESKOVEC, J.; RAJARAMAN, A.; ULLMAN, J. D. **Mining of massive datasets**. [S.l.]: Cambridge University Press, 2014.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. **Introduction to Information Retrieval**. [S.l.]: Cambridge University Press, 2009.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. **Foundations of Machine Learning**. [S.l.]: The MIT Press, 2012.

MOLNAR, S.; MORO, R.; BIELIKOVÁ, M. Trending words in digital library for term cloud-based navigation. In: SEMANTIC AND SOCIAL MEDIA ADAPTATION AND PERSONALIZATION (SMAP), 2013 8TH INTERNATIONAL WORKSHOP ON. **Anais...** [S.l.: s.n.], 2013. p.53–58.

PAGE, L. et al. **The PageRank citation ranking**: bringing order to the web. [S.l.]: Stanford InfoLab, 1999. 161–172p.

PETROVIC, S.; OSBORNE, M.; LAVRENKO, V. RT to Win! Predicting Message Propagation in Twitter. In: ICWSM. **Anais...** [S.l.: s.n.], 2011.

PROJECT, S. **SAX**. Acessado em Novembro/2016, http://www.saxproject.org/event.html.

RAJARAMAN, A. **Mining of Massive Datasets**. Cambridge: Cambridge University Press, 2011.

RAMOS, J. Using tf-idf to determine word relevance in document queries. In: OF THE FIRST INSTRUCTIONAL CONFERENCE ON MACHINE LEARNING. **Proceedings...** [S.l.: s.n.], 2003.

STEINWART, I.; CHRISTMANN, A. **Support vector machines**. [S.l.]: Springer Science & Business Media, 2008.

WANG, A. H. Detecting spam bots in online social networking sites: a machine learning approach. In: **Data and Applications Security and Privacy XXIV**. [S.l.]: Springer, 2010. p.335–342.

WU, H. C. et al. Interpreting tf-idf term weights as making relevance decisions. **ACM Transactions on Information Systems (TOIS)**, [S.l.], v.26, n.3, p.13, 2008.

ZHANG, W.; YOSHIDA, T.; TANG, X. A comparative study of TF* IDF, LSI and multi-words for text classification. **Expert Systems with Applications**, [S.l.], v.38, n.3, p.2758–2765, 2011.

ZUBIAGA, A. et al. Classifying trending topics: a typology of conversation triggers on twitter. In: ACM INTERNATIONAL CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, 20. **Proceedings...** [S.l.: s.n.], 2011. p.2461–2464.