UNIVERSIDADE FEDERAL DO CEARÁ CAMPUS QUIXADÁ
Bacharelado em Sistemas de Informação

EMANUEL EDUARDO DA SILVA OLIVEIRA

WEEWS
Um sistema recomendador de notícias relacionadas apresentadas em web widget

Quixadá-CE
2016

EMANUEL EDUARDO DA SILVA OLIVEIRA

WEEWS
Um sistema recomendador de notícias relacionadas apresentadas em web widget

Monografia apresentada ao curso de Sistemas de Informação da Universidade Federal do Ceará Campus Quixadá, como requisito para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Regis Pires Magalhães

Quixadá-CE
2016

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca do Campus de Quixadá

S45w Oliveira, Emanuel Eduardo da Silva
Weews: um sistema recomendador de notícias relacionadas apresentadas em web widget/ Emanuel Eduardo da Silva Oliveira. – 2016.
39 f. : il. color., enc. ; 30 cm.

Monografia (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Bacharelado em Sistemas de Informação, Quixadá, 2016.
Orientação: Prof. Me. Regis Pires Magalhães
Coorientação: Prof. Me. Lívio Antônio Melo Freire
Área de concentração: Computação

1. Jornais eletrônicos 2. Sistema de Recomendação 3. Algorítmo 4. Web widget I. Título.

CDD 005

EMANUEL EDUARDO DA SILVA OLIVEIRA

WEEWS
Um sistema recomendador de notícias relacionadas apresentadas em web widget

Monografia apresentada ao curso de Sistemas de Informação da Universidade Federal do Ceará Campus Quixadá, como requisito para obtenção do título de Bacharel em Sistemas de Informação.

Aprovado em: _____/fevereiro/2016

BANCA EXAMINADORA

_____
Prof. Regis Pires Magalhães

_____
Prof. Lívio Antônio Melo Freire

_____
Prof[a]. Lívia Almada Cruz

À Ilanna Cabral, minha namorada e maior amiga.
À Ilma Cabral, minha segunda mãe.
À Lúcia Bernardo, minha mãe e fonte de motivação.
Aos motoristas, comerciantes, secretários(as), moto-taxistas e colegas que me ajudaram de alguma forma.

RESUMO

As facilidades da Internet, bem como seu alcance, promoveram uma mudança cultural na forma de ler notícias. Do papel impresso para os portais, milhares de notícias são publicadas diariamente e estão disponíveis a poucos passos. Essa quantidade massiva de informação sobrecarrega os usuários, que apenas desejam encontrar algo interessante para ler. Esse desafio, de entregar somente o que é útil aos usuários, encorajou os portais a usarem alguma forma de recomendação, através de um agente filtrador que selecione o que se julga útil para o usuário. Entretanto, muitos portais optam por recomendações elaboradas manualmente, que são mais propícias à falha e requerem constantes intervenções para manter o conteúdo atualizado. Para apresentar uma alternativa às atuais formas de recomendação, esse trabalho descreve o Weews, um sistema recomendador de notícias. O Weews objetiva recomendar notícias com base nas similaridades entre elas e na relevância dos termos extraídos dos textos das notícias em um determinado período. Para comprovar a efetividade do Weews, foram realizados experimentos e análises empíricas sobre os resultados gerados. Através dessas análises, pode-se validar o algoritmo de recomendação do Weews, provando-o ser uma alternativa eficiente às formas de recomendação atuais.

Palavras-chave: Portais. Notícias. Recomendação. Algoritmo.

ABSTRACT

Internet facilities as well as its scope, caused a cultural change in the way of reading news. From printed paper to the portals, thousands of news are posted daily and are available just steps. This massive amount of information overwhelm users, who just want to find something interesting to read. This challenge, of delivering only is helpful to users, encouraged the portals to use some form of recommendation through a filtering agent who select what is useful to the user. However, many portals choose to recommendations made by hand, which are more conducive to failure and require constant intervention to keep the content always updated. To introduce an alternative to current forms of recommendation, this work describes the Weews a news recommender system. The Weews aims recommend news based on the similarities between them and the relevance of the extracted terms of the news texts in a given period. For verifying the effectiveness of Weews, experiments and empirical analysis of the results generated were performed. Through these analyzes, we can validate the Weews recommendation algorithm, proving it to be an efficient alternative to current forms of recommendation.

Keywords: Portals. News. Recommendation. Algorithm.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O ato de ler uma notícia mudou com os avanços das tecnologias e da grande rede mundial. Da tradicional leitura em papel impresso para a tela do *desktop*, *tablet* e *smartphone*, a pessoas migraram a forma de consumir esse tipo de conteúdo. Essa mudança causou um impacto cultural. Dados publicados em 2010 pelo *The New York Times*[1] já apontavam para a queda na circulação de jornais impressos. As facilidades para publicar um conteúdo na Internet, como uma notícia, beiram o simplismo. Além disso, promovem um alcance maior de usuários. Mesmo que a qualidade do conteúdo e a tradição sejam postas em questão, a forma como os meios eletrônicos de publicação vêm proporcionado o aumento do número de leitores é inquestionável. Impulsionados pelo crescente número de usuários e pelas facilidades da publicação na Internet, não demorou para que mais portais surgissem e para que o número de notícias crescesse exponencialmente e com uma diversidade de conteúdos não antes vista.

Esse cenário aconteceu naturalmente para diversos ramos do mercado. Além de notícias, produtos, livros, filmes, etc., migraram para Internet e perceberam o aumento de usuários e de itens para serem vendidos, lidos e assistidos. Um problema crítico para os portais de notícia é o volume de artigos que podem sobrecarregar os leitores. O desafio é ajudar os usuários a encontrar uma notícia interessante para ler. Para suportar tantos itens e atender tantos usuários, empresas começaram a aderir Sistemas de Recomendação (SR) para os seus sites e portais, uma vez que esse tipo de tecnologia atua como agente filtrador, entregando o que julga-se útil para os usuários. Os portais de notícias, normalmente, oferecem dois tipos de recomendação de notícias: nas páginas principais (*home page* e páginas dos canais de esporte, economia, educação, etc.) e na página individual de cada notícia, geralmente na parte inferior, na seção *"veja também"* ou *"leia mais"*. Entretanto, essas recomendações muitas vezes são elaboradas manualmente, o que exige esforço e intervenção constante para manter assuntos atualizados e na linha editorial do portal.

Existem diversos modelos de SR. Entretanto, nem todos são facilmente adaptados para determinado domínio. No domínio de notícias, o SR deve priorizar a natureza do tipo de item recomendado, uma vez que notícias possuem um curto ciclo de vida. Rapidamente uma notícia é substituída por outra que atualiza seu conteúdo ou seu assunto se torna obsoleto e não atrai mais o interesse coletivo. SR baseado em modelos, por exemplo, não se adequariam imediatamente,

[1] http://www.nytimes.com/2010/04/27/business/media/27audit.html

visto que o ciclo de vida das notícias são rapidamente iniciados e encerrados e atualizações dos modelos são considerados processamentos mais custosos.

Como uma alternativa as formas de recomendação usadas atualmente, o **Weews: sistema recomendador de notícias para provedores de conteúdo apresentado como web widget** (Weews é uma junção de *Widget* e *News*) se propõe a recomendar notícias de forma automática. Para tal, o Weews possui, como principal contribuição, um algoritmo de recomendação baseado em conteúdo que usa notícias de diversos portais para computar *hot topics* e ponderá-los à similaridade das notícias relacionadas. Além do seu algoritmo, outra contribuição do Weews é sua arquitetura distribuída, composta por dois componentes: (i) o painel administrador **Hermes**, onde os portais podem configurar a exibição do *web widget*; e (ii) o **Procedimento Recomendador**, responsável por computar as recomendações de notícias relacionadas. Como resultado, o Weews recomenda k notícias relacionadas para cada notícia de um portal, selecionando-as com base no índice de similaridade e relevância obtido na execução do algoritmo.

Esse trabalho está divido nas seguintes seções: a Seção 2 descreve os principais conceitos usados no trabalho, bem como sua relação com o Weews; a Seção 3 apresenta alguns trabalhos relacionados a SR de notícias (SRN) baseados em conteúdo; a Seção 4 descreve as contribuições do Weews: o seu algoritmo de recomendação e a sua arquitetura; a Seção 5 descreve o experimento e os resultados, bem como a análise desses resultados; e a Seção 6 conclui e pincela sobre os trabalhos futuros para tornar o Weews um SRN usado em larga escala entre os provedores de conteúdo.

2 FUNDAMENTAÇÃO TEÓRICA

Os conceitos a seguir são inerentes ao Weews e fundamentais para o entendimento dos termos usados no decorrer do trabalho. Essa seção inicia com a definição de SR, passando pela suas características e sua classificação. Posteriormente, aborda-se o tipo de SR baseado em conteúdo e o caso específico de SRN. É dedicado um tópico sobre o problema de *cold-star* e, por último, sobre medidas de similaridade.

2.1 Sistemas de Recomendação

Sistemas de Recomendação (SR) são conjuntos de ferramentas e técnicas que fornecem sugestões de itens que venham a ser úteis para os usuários (RICCI; ROKACH; SHAPIRA, 2011). *Item* é o termo usado para denotar o objeto ou o tipo de objeto recomendado pelo SR. Normalmente, os SR é projetado para um tipo específico de item, direcionando seus componentes visuais e arquiteturais para fornecer sugestões úteis e eficazes dele.

As recomendações podem ser classificadas como personalizadas ou não-personalizadas. Chama-se personalizadas as recomendações que se moldam ao perfil de um usuário ou de um grupo de usuários. Nessa abordagem de recomendação, diferentes usuários recebem diferentes sugestões. Há também as recomendações não-personalizadas, cuja a sugestão dos itens não obedece perfis e históricos específicos de consumo e de comportamento. Recomendações não-personalizadas são mais simples de desenvolver e, comumente, são apresentadas como uma *top list* ou ”mais populares”.

SR coleta informações de preferências dos usuários em relação a um conjunto de itens (BOBADILLA et al., 2013). Essas informações são obtidas de forma explícita ou implícita. De forma explícita quando os usuários pontuam os itens, e.g, uma avaliação 5 estrelas de um filme. De forma implícita quando o sistema mapeia os comportamentos dos usuários, registrando, por exemplo, quais músicas eles escutam, quais filmes que eles assistiram ou quais *websites* eles visitaram.

Para gerar as recomendações, os SR devem levar em consideração os seguintes pontos (não necessariamente todos):

1. O tipo de dado disponível no banco de dados: *ratings*, informações demográficas sobre

usuários, características e conteúdo dos itens, relacionamentos sociais entre os usuários, informações dependentes de local/contexto, etc.

2. O algoritmo de filtragem usado: demográfico, baseado em conteúdo, filtragem colaborativa, baseado em informações sociais, dependentes de contexto, híbrido, etc.

3. O modelo escolhido: baseado no uso direto dos dados (baseado em memória) ou baseado em um modelo gerado com tais dados (baseado em modelo).

4. As técnicas aplicadas: abordagens probabilísticas (*Naive Bayes*), redes bayesianas, algoritmo dos vizinhos mais próximos (*KNN*), algoritmos baseado em modelos biológicos, tais como redes neurais (*Neural Networks*) e algoritmos genéticos, modelos fuzzy, técnicas de decomposição em valores singulares (*SVD*) para reduzir o nível de dispersão, etc.

5. O nível de dispersão dos dados e a escalabilidade desejada.

6. Performance do sistema: consumo de tempo e de memória.

7. A qualidade desejada nos resultados: novidade, cobertura, precisão, etc.

As funções internas de um Sistema de Recomendação são caracterizadas pelo algoritmo de filtragem. Uma classificação amplamente utilizada divide os algoritmos de filtragem em:

- **Baseados em conteúdo**: recomendações baseadas em conteúdo são efetuadas com base nas escolhas passadas dos usuários, bem como nas análises de conteúdo dos itens que podem ser recomendados, como o texto de uma notícia ou os meta-dados de uma música. Dessas análises, uma similaridade pode ser estabelecida, que servirá como base para recomendar itens semelhantes aos itens que se relacionam com o perfil de um usuário, seja por uma avaliação ou por pertencer ao seu histórico de consumo.

- **Colaborativos**: diferente da filtragem baseada em conteúdo, os algoritmos de filtragem colaborativa predizem a utilidade dos itens para um usuário baseando-se nos itens previamente avaliados por outros usuários. A motivação para a filtragem colaborativa vem da ideia de que as pessoas muitas vezes recebem melhores recomendações de alguém com gostos semelhantes aos seus.

- **Híbridos**: combinam mais de um algoritmo de filtragem para explorar méritos de cada um. Vários sistemas de recomendação usam uma abordagem híbrida através da combinação

de filtragem colaborativa e baseada em conteúdo, o que ajuda a evitar certas limitações dos sistemas baseados em conteúdo, como a fragilidade para explorar a diversidade, e dos sistemas colaborativos, como o *cold-start*.

O objetivo do SR é reduzir o problema de estimar a avaliação de um item ainda não usado pelo usuário (ADOMAVICIUS; TUZHILIN, 2005). Essa estimativa é normalmente baseada nas avaliações desse usuário para outros itens ou de outros usuários sobre o item em questão, ou sobre outros itens. Uma vez que se pode estimar as avaliações para os itens ainda não avaliados, é possível recomendar ao usuário o(s) item(s) com a(s) mais alta(s) avaliação(ões) estimada(s).

Uma formulação do problema da recomendação é: seja C o conjunto de todos os usuários e I o conjunto de todos os itens que podem ser recomendados. Seja u uma função utilitária que mede a utilidade de um item i para um usuário c, isto é, $u : C \times I \rightarrow R$, onde R é são inteiros não negativos ou números reais dentro de um intervalo, que representam a utilidade dos itens para os usuários. Então para cada usuário $c \in C$, queremos escolher um item $i_c \in I$ que maximize a utilidade para o usuário. Mais formalmente:

$$\forall c \in C, i_c = \underset{i \in I}{\operatorname{argmax}}\, u(c, i) \tag{2.1}$$

Como a proposta desse trabalho utiliza puramente um sistema de recomendação baseado em conteúdo e já foram dadas as devidas considerações aos demais tipos de sistema de recomendação, os tópicos subsequentes cobrem características sobre recomendação baseado em conteúdo.

2.1.1 Recomendação baseada em conteúdo

Devido aos significantes e recentes avanços no campo de recuperação de informação e à importância de várias aplicações textuais, cresce o número SR baseados em conteúdo que recomendam itens documentos, como notícias e *web pages*.

Recomendação baseada em conteúdo é uma tecnologia em resposta ao desafio da sobrecarga de informação em geral. Com base em um perfil de interesse e preferências do usuário, o SR recomenda itens que possam ser de interesse ou valor para o usuário (LIU; DOLAN; PEDERSEN, 2010). Métodos baseados em conteúdo desempenham um papel importante no SR, uma vez que são capazes de recomendar itens que ainda não foram avaliados, atenuando o SR do

problema de *cold-start* (Seção 2.2)

SR baseado em conteúdo constrói perfis de usuários e calcula a utilidade (Função 2.1) com base na similaridade entre o conteúdo de um item e o perfil de um usuário. Há várias maneiras de representar os atributos de um item e os perfis de um usuário. Em se tratando de documentos textuais, uma maneira simples de representar os atributos de um item é através da abordagem "saco de palavras"(*bag of words*), na qual os termos do documento, ponderados de algum modo, constituem os atributos e o perfil de um usuário é representado pelo conjunto dos termos dos documentos lidos por ele por exemplo. Dessa forma, para obter o *bag of words* e a representação computacional ideal para computar a utilidade dos documentos para o usuário, é necessário efetuar um processamento textual sobre o conteúdo dos documentos.

Considere um SR baseado em conteúdo onde os itens recomendados nesse sistema são primordialmente textuais (documentos, como *web pages* ou *e-books*). Seja I o conjunto de documentos nesse sistema. Os termos de um documento são obtidos através de mineração de dados textuais e técnicas de processamento de linguagem natural (*NLP*). Para obter os atributos textuais (termos) de um documento, normalmente, submete-se o texto a uma sequência de tarefas de pré-processamento. São tarefas de um pré-processamento:

1. Tokenização: recebe como entrada um texto e produz uma sequência de termos não vazios (*tokens*). De maneira simplificada, o texto é quebrado por espaços e os elementos resultantes geram tokens quando possuem caracteres comuns de palavras ou números, ou são divididos em mais de um termo quando iniciam ou terminam por pontuação.

2. Reconhecimento de entidades Nomeadas: identifica os termos que são nomes próprios. Os termos que iniciam por caracteres maiúsculos são candidatos a nomes próprios. Procura-se encontrar sequências de candidatos que se referem a apenas uma entidade ou identificar quando um candidato que inicia uma sentença não é nome próprio. O passo seguinte é classificar as entidades considerando um conjunto de categorias (em geral, *pessoa*, *lugar*, *organização*, *outro*).

3. Remoção de *stopwords*: recebe uma sequência de tokens e remove os termos que são muito comuns (*stopwords*) no idioma ou no domínio. Essas palavras possuem pouco valor semântico e não são úteis para discriminar o texto. A lista de *stopwords* é um parâmetro do procedimento.

4. Redução ao radical: recebe um termo e remove a varição gramatical por conjugação, gênero, número, etc., resultando apenas o radical.

5. Normalização: transforma os caracteres dos tokens para maiúsculo ou minúsculo. Outras tarefas comuns a essa etapa incluem a remoção de caracteres especiais e números.

O conjunto de termos dos itens textuais (documentos), resultantes após a aplicação de uma combinação das tarefas descritas acima, forma o vocabulário V. No modelo espaço-vetorial, o documento $i \in I$ é representado através do vetor $w_i = \{w_{i1}, w_{i2}, \ldots, w_{i,|V|}\}$, em que o peso w_{ij} estabelece a relação entre o termo j do vocabulário e o documento i.

Os pesos atribuídos aos termos são computados através de uma gama de algoritmos. Esses algoritmos usam e geram informações quantitativas sobre os termos, como frequência e relevância. São alguns desses algoritmos:

- *Term Frequency * Inverse Document Frequency (TF*IDF)*: o peso do termo é obtido pelo produto da frequência do termo em um documento e a frequência inversa do número de documentos em que esse termo ocorre (ROCCHIO, 1971). Essa medida atribui alto peso ao termo que é muito frequente em um documento e é raro no corpus de documentos, e baixo peso ao termo que aparece em muitos documentos do corpus. Quando certas palavras ocorrem em vários documentos do corpus D, em algumas aplicações, considera-se que elas não são relevantes para um documento em particular.

- *Term Frequency * Proportional Document Frequency (TF*PDF)*: Diferente do *TF*IDF*, no algoritmo *TF*PDF*, o peso do termo de um canal é linearmente proporcional à frequência do termo dentro do canal, e exponencialmente proporcional à razão de documentos no canal que contém esse termo. O peso total de um termo resulta da soma dos pesos dele em cada canal (BUN; ISHIZUKA, 2002), como demonstrado nas Equações 2.2 e 2.3.

$$W_j = \sum_{c=1}^{c=D} |F_{jc}| \exp(\frac{n_{jc}}{N_c}), \tag{2.2}$$

$$|F_{jc}| = \frac{F_{jc}}{\sqrt{\sum_{k=1}^{k=K} {F_{kc}}^2}} \tag{2.3}$$

Onde W_j = Peso do termo j; F_{jc} = Frequência de termo j no canal c; n_{jc} = Número de documentos do canal c onde o termo j ocorre; N_c = Número total de documentos no canal c; k = número total de termos em um canal; D = número de canais.

- *Normalized Term Frequency (NTF)*: Em certas aplicações, é necessário adotar pesos que ponderem a relevância do termo no documento. No esquema de pesos *frequência normalizada*, o valor de $w_{ij} = tf_{ij}$ é calculado de acordo com a Equação 2.4.

$$tf_{ij} = 0.5 + 0.5 \times \frac{f_{ij}}{max\{f_{ik} : k \in i\}} \tag{2.4}$$

  Onde f_{ij} é a frequência do termo j no documento i. Nesse modelo, palavras com frequência maior são favorecidas.

Os métodos de atribuição de pesos usados nesse trabalho são: (i) *TF-PDF*, usado na atribuição de global de pesos, que compreende o conjunto de termos do vocabulário formado a partir do conjunto de documentos (notícias), e (ii) *NTF* para a atribuição de pesos locais, compreendendo o conjunto de termos do documento. O *TF-IDF* não é usado neste trabalho, pois o algoritmo de recomendação em questão pondera o cálculo de relevância de um documento com base no conceito de tópicos quentes (*hot topics*). Portanto, precisa-se de uma medida que valorize a ocorrência dos termos em muitos canais e documentos, o inverso que o *TF-IDF* propõe.

2.1.2 Recomendação de Notícias

Sistemas de Recomendação de Notícias (SRN) são casos específicos de SR. Essa especificidade existe por características particulares das notícias, como o ciclo de vida curto. Após alguns dias ou horas, uma notícia é substituída por outra que atualiza seu conteúdo ou perde relevância por tratar de tópicos que não despertam mais interesse coletivo. Essa característica diferencia um SRN de um SR de produtos por exemplo, pois os produtos podem ser adquiridos futuramente, não se configurando em uma desvantagem para um usuário ou cliente, como ler uma notícia não atual. Outra especificidade é relacionada ao perfil de usuário que acessa o portal de notícias. Muitos usuários são visitantes casuais e acessam o portal de muitos dispositivos e sem nenhum cadastro prévio, dificultando a detecção de comportamento ou a coleta de outras informações, como dados demográficos.

Os provedores de conteúdo inserem conteúdo novo a todo instante procurando cobrir tópicos que julgam despertar interesse de seus leitores. Os editores dos portais de notícias

selecionam o que vai ser publicado respeitando a linha editorial do veículo de comunicação, que determina os tópicos a serem abordados pelas notícias. Os artigos produzidos cobrem: (i) conteúdo factual, referente a tópicos quentes do período que atraem um público grande, ou (ii) atemporal, sobre tópicos especializados e destinado a uma audiência específica.

Os portais seguem um fluxo padrão e disseminado de recomendação, que ocorre de duas formas. Na primeira forma, são apresentadas as últimas e mais relevantes notícias na página inicial do portal (Figura 1), onde o grau de relevância é manualmente calculado pelos provedores de conteúdo online. A outra forma ocorre na página individual de uma notícia, onde são recomendadas notícias em seções intituladas como *"Veja também"* ou *"Notícias Relacionadas"*, geralmente nas partes inferiores das páginas (Figura 2).

Figura 1 – Página inicial com recomendações gerais.

Fonte: captura do UOL alterada digitalmente

2.2 *Cold-start*

Cold-start é a situação em que um SR é incapaz de fazer recomendações significativas devido a uma falta inicial de informações sobre os usuários (SCHAFER et al., 2007). O *cold-start* pode acontecer de três formas: para uma nova comunidade, para um novo usuário e para um novo item.

O *cold-start* acontece em novas comunidades quando o SR está iniciando sua atividade. Nesse estágio, o SR não possui informações suficientes sobre os históricos de consumo dos usuários e suas avaliações sobre itens para efetuar recomendações eficientemente. Nor-

Figura 2 – Recomendações de notícias na página individual de uma notícia

Fonte: captura do UOL alterada digitalmente

malmente, duas propostas são usadas para tratar esse problema: (a) encorajar os usuários a realizarem avaliações sobre itens ou (b) realizar recomendações somente quando o SR possuir informações suficientes sobre usuários e suas avaliações.

Para novos itens, o *cold-start* acontece devido a ausência de avaliações iniciais, já que os itens foram inseridos recentemente no sistema. Eventualmente, esses itens correm o risco de não serem recomendados e se tornarem desconhecidos para a maioria dos usuários, que não avaliam o item por não tomarem conhecimento da existência dele. Em alguns contextos, como em recomendação de filmes, o *cold-start* para novos itens causa menos impacto do que em outros contextos, como recomendação de notícias, pois os itens do primeiro contexto podem ser encontrados de outra forma posteriormente (BOBADILLA et al., 2013). Para amenizar o *cold-start* para novos itens, o SR confia a um grupo de usuários motivados a tarefa de avaliar cada novo item que entra no sistema.

Um dos maiores desafios para sistemas de recomendação é o *cold-start* para novos usuários. Novos usuários são escassos de avaliações, logo o SR não consegue recomendar

eficientemente recomendações personalizadas para eles. O uso de filtragem colaborativa pura não vai ajudar o sistema a sanar o problema de recomendar para um novo usuário (SCHEIN et al., 2002). A maioria das tentativas de aliviar o *cold-start* para novos usuários se resumem em aproveitar ao máximo as informações existentes sobre o perfil do usuário e usar recomendação baseada em conteúdo.

O SR de recomendação desse trabalho é puramente baseado em conteúdo, logo ele supera o problema de *cold-start*, pois não usa dados para personalização de recomendações para os usuários. Um eventual problema de *cold-start* que ameaçaria o SR desse trabalho, seria do tipo *nova comunidade*, visto que, no início, o SR não possuirá uma quantidade de notícias essencial para aprimorar seus resultados, já o algoritmo de atribuição de pesos *TF-PDF* tem a precisão diretamente proporcional à quantidade de documentos e canais usados no método(BUN; ISHIZUKA, 2002). Uma solução é acionar processos de coleta, executados por *crawlers*, nas páginas dos portais de notícias, antes mesmo de publicar o serviço de recomendação. Assim, o SR já possuirá uma quantidade razoável de documentos disponíveis para o algoritmo recomendar eficientemente.

2.3 Medidas de similaridade

Medidas de similaridade são funções que quantificam a similaridade entre dois itens, inclusive dois usuários. Tem sentido inverso à definição de medidas de distância, ao passo que essas funções atribuem altos valores de imagem para itens dissimilares. As medidas de similaridade são utilizadas geralmente para estabelecer agrupamentos ou atribuição de pesos para fins de comparação. São medidas de similaridade conhecidas: Euclidiana, Coeficiente de Correlação de Pearson, Erro Quadrático Médio, Similaridade Jaccard e Similaridade Cosseno.

O conceito de similaridade é abordado a seguir sobre a definição das duas medidas de similaridade usadas no trabalho, *Jaccard* e *cosseno*, e sobre a aplicação dessas medidas para o cálculo de similaridade entre dois documentos de texto. A noção de similaridade entre os textos t_1 e t_2, representados pelos vetores w_{t_1} e w_{t_2}, será usada para medir a relação entre os elementos textuais.

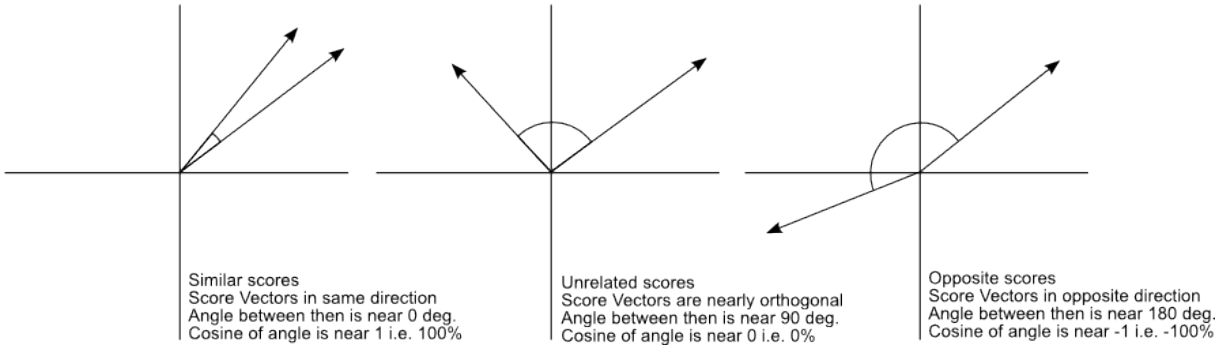
2.3.1 Similaridade cosseno

Dois documentos com conteúdos semelhantes podem ter uma diferença significativa simplesmente porque um tem mais palavras do que o outro. Assim, as distribuições relativas dos termos podem ser idênticas nos dois documentos, mas as frequências de um podem ser maiores do que em outro.

Para dois vetores de termos gerados por algoritmos de atribuição de peso, é possível utilizar a função cosseno, definida conforme a Equação 2.5, para computar a similaridade entre eles. Ao calcular a similaridade dessa forma, vetores com mesma orientação, perpendiculares ou em direções opostas no hiperplano possuem similaridade 1, 0 e -1, respectivamente (Figura 3). Vetores têm orientação similar quando apresentam componentes com valores próximos.

$$Cosine(t_1, t_2) = \frac{w_{t_1}.w_{t_2}}{|w_{t_1}|\,.\,|w_{t_2}|} \tag{2.5}$$

Figura 3 – Os valores de similaridade cosseno para diferentes documentos. 1 **(mesma direção),** 0 **(**90 **graus),** -1 **(direções opostas).**



Fonte: Machine Learning :: Cosine Similarity for Vector Space Models (Part III) - Pyevolve

Nesse trabalho, faz-se o uso da similaridade cosseno junto ao uso de dois esquemas de atribuição de pesos para os termos das notícias: (i) *TF-PDF*, do qual se obtém uma atribuição global (para todas as notícias) de pesos dos termos que compõe o vocabulário; e (ii) *NTF*, do qual se obtém uma atribuição de pesos local dos termos que compõe uma notícia. A similaridade cosseno entre os vetores global (*TF-PDF*) e o local (*NTF*) é a relevância de uma notícia. Entretanto, no cálculo de relevância de cada notícia, são usados, do vocabulário, apenas os termos que ocorrem em determinada notícia. Portanto, se uma notícia possui muitos termos relevantes localmente e esses termos também são relevantes globalmente, então ela é relevante

no conjunto de notícias.

2.3.2 Similaridade Jaccard

A similaridade Jaccard, também conhecida como índice Jaccard, é uma medida que permite avaliar a semelhança ou a diversidade entre dois conjuntos. Sejam A e B dois conjuntos. A similaridade Jaccard entre A e B é razão entre o tamanho da interseção e o tamanho da união entre A e B (Equação 2.6). A imagem da similaridade Jaccard é composta por valores reais maiores ou iguais a 0 e menores ou iguais a 1 ($0 \leq Jaccard(A, B) \leq 1$). A similaridade é mais próxima de 1 quando A e B possuem muitos elementos em comum, e mais próxima de 0 caso contrário.

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{2.6}$$

O uso da similaridade Jaccard para comparação de notícias exige uma representação adequada. Notícias são formadas por elementos textuais principalmente, como título e texto. Esses elementos textuais são compostos por sentenças e as sentenças por palavras. A ordem como as palavras e as sentenças aparecem no texto formam o sentido e as informações da notícia. A similaridade Jaccard mede a similaridade entre conjuntos, logo a ordem dos elementos no conjunto não influencia no resultado. Representar as notícias como um saco de palavras somente, tornam sentenças como *"Annali foi assaltada na padaria do Sr. Ulisses"* e *"A padaria do Sr. Ulisses foi assaltada por Annali"* muito similares comparando-as através da similaridade Jaccard, porém elas apresentam significados opostos. Uma alternativa ao saco de palavras é o uso de *k-shingles*, uma estratégia de representação comum em *Natural Language Processing* (*NLP*), na qual o documento é representado por subsequências de caracteres de tamanho k dentro do documento (RAJARAMAN et al., 2012).

Nesse trabalho, faz-se o uso da similaridade Jaccard. Para tal, as notícias são representadas pelas entidades referenciadas no texto (*pessoas*, *lugares*, *empresas*, *marcas*, etc.). As entidades do texto são obtidas por uma técnica de extração de entidades baseada na seguinte heurística: uma entidade é qualquer sequência de palavras que começam com letra maiúscula. Essa representação facilita na busca por notícias que falam sobre o mesmo assunto, uma vez que as ações dentro do texto são descartadas para tal e que a ordem das entidades, para esse fim, não é relevante.

3 TRABALHOS RELACIONADOS

Duas abordagens são comumente usadas em SR: recomendação baseada em conteúdo e filtragem colaborativa. Essa seção aborda trabalhos que se relacionam com esse trabalho por fazerem uso da recomendação baseada em conteúdo.

(PHELAN; MCCARTHY; SMYTH, 2009) propõe um SRN que usa as tendências de assuntos nos *tweets* de um usuário e um feed *RSS* (*Really Simple Syndication*), que agrega os canais preferidos do usuário. No processo de recomendação, são extraídos da *timeline* de cada usuário os termos com maiores incidências. Estes termos são utilizados para ranquear as notícias disponíveis no estado corrente do *feed RSS* do usuário, priorizando as notícias com maior ocorrência dos termos obtidos dos *tweets* na ordem de recomendação. Entretanto, (PHELAN; MCCARTHY; SMYTH, 2009) atendem apenas usuários que possuem uma conta no *Twitter*. Isso não promove o alcance da recomendação à massa de usuários casuais, que acessam portais de notícias anonimamente. O Weews, o SR desse trabalho, atende usuários diversos, cadastros ou não nos portais de notícias.

(KOMPAN; BIELIKOVÁ, 2010) também utilizam técnicas baseadas em conteúdo para recomendar notícias. Vetores de estrutura específica foram criados para representar as notícias. Cada vetor é dividido em partes e cada parte contém os termos retirados de uma notícia e os pesos desses termos. São as parte dessa estrutura: título, frequência dos termos do título no documento, categoria, palavras-chave, *named-entities* (nomes próprios) e o *CLI* (*Coleman-Liau Index*, importante na organização dos resultados). Cada parte tem seus critérios para calcular o peso de um termo. Na *frequência dos termos do título no documento*, a frequência de cada termo do título é a razão entre a quantidade de ocorrências do termo no texto do documento e o somatório das ocorrências de todos os outros termos do texto. As computações que são usadas para calcular os pesos dos termos são feitas sobre a saída de um Processamento de Linguagem Natural (*NLP*). Depois de elaborar os vetores, o sistema pode manter, para cada notícia do dataset, uma lista de notícias relacionadas, visitadas ou não, por um usuário. Essas listas são utilizadas no algoritmo de recomendação como candidatas a recomendação quando o SR buscar por correspondências do histórico de acesso da sessão do usuário.

O Weews propõe um sistema de recomendação semelhante a (KOMPAN; BIELIKOVÁ, 2010), pois cria uma lista com n notícias semelhantes para cada notícia. Além disso, o Weews aprimora essa lista, pois pondera, na seleção das notícias, a relevância de uma notícia com

base nas notícias de vários provedores de conteúdo. A relevância de uma notícia reflete o quão importante é seu conteúdo perante aos ”tópicos quentes” de vários portais.

A Tabela 1 apresenta um comparativo entre as características dos trabalhos relacionados relacionados e o Weews.

Tabela 1 – Comparativo de características entre os trabalhos relacionados e o Weews

Trabalho	Requer conta de usuário	Personalizado	Recomendação top *n*	Fonte de conteúdo
(PHELAN; MCCARTHY; SMYTH, 2009)	Sim	Sim	Sim	Twitter e RSS
(KOMPAN; BIELIKOVÁ, 2010)	Não	Não	Sim	Portal único
Weews	Não	Não	Sim	Portais variados

4 WEEWS - RECOMENDAÇÃO DE NOTÍCIAS BASEADA EM SIMILIRIDADE E RELEVÂNCIA

Como objetivo de ajudar o usuário a encontrar uma notícia interessante para ler e ajudar os portais de notícias a automatizarem o processo de recomendação de notícias relacionadas, é que o Weews se apresenta como alternativa à forma de recomendação manual, usada em alguns provedores de conteúdo. O Weews também se apresenta como uma alternativa para portais que não oferecem recomendação de notícias relacionadas.

Em muitos portais de notícias, a página inicial, que possui dezenas de notícias organizadas de maneira particular por cada portal, algumas vezes se torna visualmente desagradável e complicada para o usuário. Então, baseados em um princípio no qual se assume que o usuário prefere ler o que lhe convém ser interessante, e que, se o usuário clicou em uma notícia dentre as dezenas disponíveis, essa notícia e o assunto que ela trata têm maior probabilidade de serem do interesse dele, os portais de notícias podem recomendar notícias relacionadas à notícia na qual o usuário se encontra.

A seções seguintes apresentam o problema que o Weews se propõe a resolver e como ele é composto, organizado e executado para resolvê-lo.

4.1 Definição do Problema

Como citado, muitos portais oferecem duas formas de recomendação: uma recomendação geral na *home page* e uma mais específica na página de cada notícia, normalmente na seção *"veja também"*. Essa última apresenta dois tipos de notícias recomendas: notícias relacionadas, *i.e.*, que tratam sobre o mesmo assunto, e notícias patrocinadas (propagandas).

O Weews automatiza o processo de recomendação de notícias relacionadas, visto que as recomendações atuais tendem às linhas editoriais de cada portal e muitas vezes são criadas manualmente. Portanto, se fez necessário um sistema de recomendação que sugira, automaticamente, notícias relacionadas para cada notícia de um portal, objetivando a continuidade e a manutenção de um determinado assunto.

Mais formalmente: seja P o conjunto dos provedores de conteúdo que não efetuam recomendações de notícias relacionadas na página individual de uma notícia ou, se efetuam, as

faz manualmente. Seja N o conjunto de todas as notícias publicadas por P no período t. Seja $N_p \subset N$ o conjunto de notícias do portal $p \in P$. Para cada notícia $n_p \in N_p$, deseja-se encontrar automaticamente k notícias relacionadas e relevantes a n_p, tal que $R_{n_p} = \{n_{p1}, n_{p2}, \ldots, n_{p|k|}\}$ seja o conjunto de notícias recomendadas e R_{n_p} é um subconjunto de N_p ($R_{n_p} \subset N_p$).

4.2 Algoritmo de recomendação

O algoritmo de recomendação do Weews é classificado como **baseado em conteúdo**, pois analisa o texto dos items (notícias) a fim de obter similaridades e índices de relevância para detecção de *hot topics*. Ponderando entre a similaridade das notícias e relevância dos termos, o algoritmo consegue resolver o problema de recomendar k notícias relacionadas e relevantes para cada notícia dos provedores de conteúdo. O Algoritmo de recomendação é demonstrado no Algoritmo 1.

Data: N = news set published in period t, k = number of recommendations per news, α = weighting coefficient
Result: news with recommendations

$C \longleftarrow findChannels(\mathrm{N})$
for $c_i \in C$ **do**
  $N_{c_i} \longleftarrow findNewsByChannel(N, c_i)$
  $TFPDF_{c_i} \longleftarrow computeTfPdf(N_{c_i})$
  for $n_{C_i} \in N_{c_i}$ **do**
    $top_K \longleftarrow createPriorityQueue(k)$
    $NTF_{n_{C_i}} \longleftarrow computeNtf(aNews.text)$
    $E_{n_{C_i}} \longleftarrow extractEntities(aNews.text)$
    for $o_{c_i} \in \{N_{c_i} - \{n_{c_i}\}\} | o_{c_i}.portal \equiv n_{C_i}.portal$ **do**
      $NTF_{o_{c_i}} \longleftarrow computeNtf(o_{c_i}.text)$
      $E_{o_{c_i}} \longleftarrow extractEntities(o_{c_i}.text)$
      $relevance_{o_{c_i}} \longleftarrow cosineSimilarity(TFPDF_{c_i}, NTF_{o_{c_i}})$
      $similarity_{n_{c_i}\, o_{c_i}} \longleftarrow jaccardSimilarity(E_{n_{C_i}}, E_{o_{c_i}})$
      $w_{n_{c_i}\, o_{c_i}} \longleftarrow (similarity_{n_{c_i}\, o_{c_i}} \times \alpha) + ((1-\alpha) \times relevance_{o_{c_i}})$
      $topK.enqueue(o_{c_i}, w_{n_{c_i}\, o_{c_i}})$
    end
    $n_{c_i}.recommendationList \longleftarrow top_K$
  end
end

Algorithm 1: Algoritmo de recomendação de notícias relacionadas e relevantes do Weews

O Algoritmo 1 recebe 3 argumentos como entrada: o conjunto N das notícias publicadas pelos provedores de conteúdo (portais de notícias) em um período t; um inteiro k que representa o tamanho das recomendações para cada notícia; e o fator de ponderação $\alpha \in \mathbb{R} \mid 0 \leq \alpha \leq 1$. A execução é mais detalhada na Subseção 4.3.2. Entretanto, cabe aqui descrever alguns procedimentos omitidos no Algoritmo 1 (omitidos para não deixar a listagem do algoritmo tão extensa). São eles:

1. A função $findChannels$ (linha 1) encontra o conjunto de canais C (categorias, como esporte, economia e educação) dentro do conjunto de notícias N.

2. A função $findNewsByChannel$ (linha 3) encontra o subconjunto de notícias $N_{c_i} \subset N$ das notícias que pertencem ao canal $c_i \in C$.

3. A função $computeTfPdf$ (linha 4) recebe o conjunto de notícias N_{c_i} e obtém um vector $TFPDF_{c_i} = \{(t_1, w_1), (t_1, w_1), \ldots, (t_n, w_n)\}$, onde t_i é um termo do conjunto de termos obtidos do pré-processamento das notícias de N_{c_i} e w_i é o seu índice ***TF-PDF***, representando a relevância do termo t_i no canal c_i.

4. A função $createPriorityQueue$ (linha 6) cria uma fila de prioridade ordenada pelo índice de similaridade e relevância $\forall n_{c_i} \in N_{c_i}$. Essa fila de prioridade é lista das top k notícias mais semelhantes e relevantes para cada notícia.

5. A função $computeNtf$ (linhas 7 e 10) obtém, para cada notícia n_j, o vetor $NTF_{n_j} = \{(t_1, w_1), (t_1, w_1), \ldots, (t_n, w_n)\}$, onde t_i é um termo do conjunto de termos obtidos do pré-processamento da notícia n_j, e w_i é o seu índice ***NTF***, representando a relevância de t_i para a notícia $n_j \in c_i$.

6. A função $extractEntities$ (linhas 8 e 11) obtém, para cada notícia n_j, o conjunto $E_{n_j} = \{e_1, e_2, \ldots, e_n\}$, onde e_i é uma entidade (*pessoa*, *lugar*, *empresa*, etc.) extraída através da heurística que considera como entidade a sequência máxima de palavras que começam com letra maiúscula.

7. A linha 12 contém o cálculo de similaridade entre o vetor $TFPDF_{c_i}$ dos termos do canal c_i e o vetor $NTF_{o_{c_i}}$ dos termos da notícia o_{c_i}, que é uma notícia do mesmo canal (c_i) e portal da notícia n_{c_i}. Esse cálculo considera apenas a interseção dos termos nos dois

vetores ($TFPDF_{c_i} \cap NTF_{o_{c_i}}$), uma vez que o vetor $TFPDF_{c_i}$ contém todos os termos do vocabulário, o tornando maior do que o vetor $NTF_{o_{c_i}}$.

8. A linha 13 contém o cálculo de similaridade entre o conjunto $E_{n_{c_i}}$ das entidades da notícia n_{c_i} e o conjunto $E_{o_{c_i}}$ das entidades da notícia o_{c_i}, que é uma notícia do mesmo canal (c_i) e portal da notícia n_{c_i}.

9. Na linha 14, o algoritmo obtém o índice de similaridade e relevância de o_{c_i} para n_{c_i}, ponderando-o, através de do coeficiente de ponderação α, entre a relevância global da notícia o_{c_i} ($relevance_{o_{c_i}}$) e a similaridade entre o_{c_i} e n_{c_i} ($similarity_{n_{c_i}\, o_{c_i}}$). Por exemplo, tome como estado: $similarity_{n_{c_i}\, o_{c_i}} = 0.5$, $relevance_{o_{c_i}} = 0.7$ e $\alpha = 0.9$. Logo, o índice de similaridade e relevância $w_{n_{c_i}\, o_{c_i}} = 0.52$.

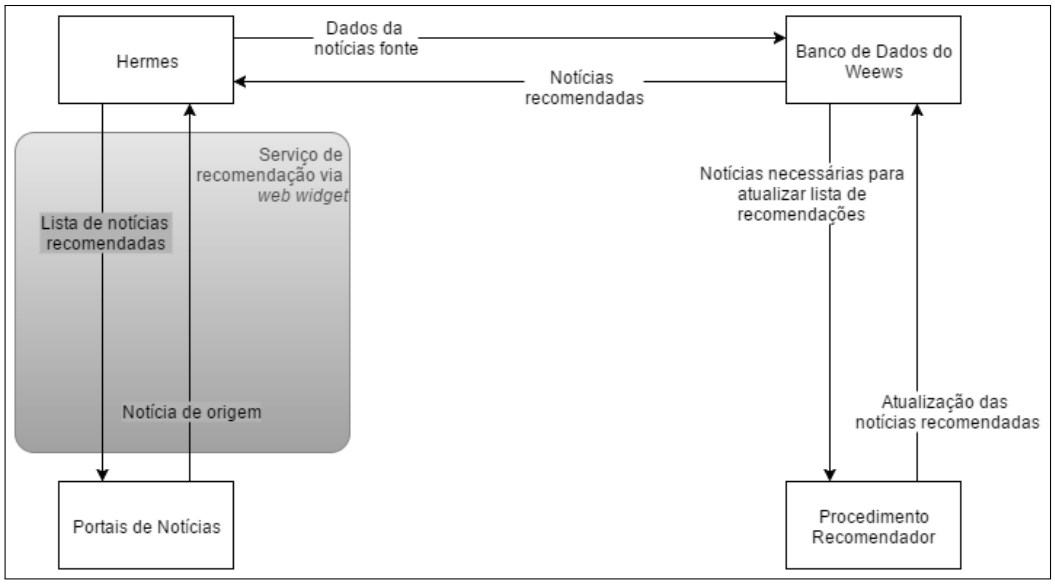
10. Na linha 15, o algoritmo insere a notícia o_{c_i}, junto ao índice de similaridade e relevância atribuído a $w_{n_{c_i}, o_{c_i}}$, na fila de prioridade top_K.

11. Na linha 17, o algoritmo atribui à notícia n_{c_i} as k notícias mais semelhantes e relevantes à ela, obedecendo a prioridade do índice $w_{n_{c_i}\, o_{c_i}}$

4.3 Arquitetura

O Weews é um sistema que oferece um serviço de recomendação para portais de notícia. As recomendações do Weews são materializadas através de um *web widget* que deve ser acoplado nas páginas das notícias obedecendo as configurações definidas pelos portais. O Weews é composto por: (i) o painel administrador **Hermes**, onde os portais de notícias configuram as preferências do seu *web widget*; e (ii)o procedimento de computação das recomendações, ou **Procedimento Recomendador**. A Figura 4 apresenta a arquitetura do Weews e como os seus principais componentes se comunicam.

Figura 4 – Principais componentes do Weews



Fonte: autoria própria

4.3.1 Painel Administrador Hermes

Para um portal consumir o serviço do Weews, uma pessoa responsável por ele deve acessar a sua conta no Hermes[1], o painel administrador do Weews, e configurar as preferências do *web widget*. Existem três tipos de configurações no Hermes:

- Confugurações de conta: nesse tipo de configuração, é possível modificar dados de acesso à conta, como e-mail e senha.
- Configurações de exibição: correspondem às configurações da exibição do *web widget* nas página das notícias do portal, como o título do *web widget*, o número de notícias que serão recomendadas e um arquivo de estilos (CSS).
- Configurações de coleta: para o Weews computar suas recomendações, ele precisa coletar dados sobre as notícias nas quais o *web widget* será usado. São dados sobre as notícias coletados pelo Weews: canal editorial (esporte, economia, etc.), título, data de publicação, descrição (opicional), imagem principal (opicional) e texto. Para coletar esses dados da página de cada notícia, o Hermes solicita ao usuário que representa o portal, seletores de identificação de elementos (no mesmo formato de seletores CSS[2]).

[1] http://weews.azurewebsites.net/
[2] http://www.w3schools.com/cssref/css_selectors.asp

A Figura 5 apresenta um exemplo de quais seletores são usados na página de uma notícia. Nesse exemplo, foram omitidos os seletores da descrição e da imagem, visto que a página não possui esses elementos. Por exigir um pouco de domínio técnico, o Hermes requer que o representante do portal de notícias tenha conhecimento suficiente para configurar essas preferências. Normalmente, esse representante pertence à área de TI do portal.

Figura 5 – Exemplo dos seletores na página de uma notícia do portal UOL

Fonte: captura do UOL alterada digitalmente

Para acoplar o *web widget* no portal de notícias, o Hermes fornece uma URL de acesso para consumo. Através dessa URL, o portal requisita o serviço de recomendação do Weews, informando apenas a URL de origem, que é o endereço da notícia na qual o *web widget* será

exibido. A Figura 6 exibe como ficaria o *web widget* do Weews em uma notícia do portal G1[3].

Figura 6 – Exemplo de uso do *web widget* no portal de notícias G1

Fonte: captura do G1 alterada digitalmente

4.3.2 Procedimento Recomendador

Com o objetivo de evitar um gargalo na eficiência das recomendações, foi decidido que o algoritmo de recomendação seria executado assincronamente, em outro processo, restando ao *web widget* apenas consumir o resultado dos algoritmos persistidos em um banco de dados. O Procedimento Recomendador é um programa executado em intervalos fixos de tempo e sempre que uma nova notícia é inserida na base de dados do Weews.

O Procedimento Recomendador recebe (recupera) como entrada uma lista de todas as notícias publicadas nos últimos 7 dias. Essa quantidade foi heuristicamente selecionada, pois apresenta um prazo razoável para assuntos quentes. Entretanto, ela pode ser facilmente parametrizada. No processamento, o Procedimento Recomendador efetua uma série de operações:

[3]http://tinyurl.com/g1-weews-web-widget

1. Pré-processamento de texto: para cada notícia, são realizadas sobre o seu texto, algumas tarefas de pré-processamento de texto: tokenização, remoção de stopwords e redução ao radical. Ao final dessa etapa, cada notícia é representada por um conjunto de *stems*, que são as raízes das palavras que não são *stopwords*.

2. Detecção de entidades: ainda sobre o texto original, é realizado o reconhecimento de entidades. Ao final dessa etapa, é obtido, para cada notícia, um conjunto de entidades.

3. Cálculo global de pesos: através da saída do pré-processamento é possível utilizar o algoritmo *TF-PDF* para computar o vetor global de relevância dos termos. Esse vetor é composto pelos termos (*stems*) do vocabulário global, que é a união de todos os conjuntos de termos das notícias de um canal, e seus respectivos pesos obtidos através do algoritmo *TF-PDF*.

4. Cálculo local de pesos: através do resultado do pré-processamento é possível utilizar o algoritmo *NTF* para computar o vetor local de relevância dos termos. Esse vetor é composto pelos termos (*stems*) do texto de cada notícia e seus respectivos pesos obtidos através do algoritmo *NTF*.

5. Cálculo de relevância global: para cada notícia, é obtida sua relevância em relação vetor global do vocabulário. Essa relevância é obtida através da medida similaridade cosseno entre o vetor local da notícia(*NTF*) e o vetor global do vocabulário(*TF-PDF*). Se os termos de uma notícia forem relevantes localmente e globalmente, essa notícia receberá um alto valor de relevância.

6. Cálculo de similaridade entre notícias: para as notícias de um mesmo portal e canal, são computadas suas as similaridades através da medida de similaridade Jaccard. Como entrada, a similaridade jacard recebe o conjunto de entidades de duas notícias e atribui valores próximos a 1 caso as notícias compartilhem muitas entidades, ou valores próximos a 0 caso contrário.

7. *Ranking* top K por notícia: cada notícia recebe uma lista de N notícias relacionadas, concretizando o resultado da recomendação. K é uma **configuração de exibição** realizada no Hermes por cada portal de notícia. Para montar essa lista por notícia, o Procedimento Recomendador executa, uma rotina de atribuição de índices de relevância-semelhança para notícias dos mesmo portal e canal. Esse índice é computado pela função 4.1, uma

média ponderada entre a relevância global de uma notícia (N_i) e a similaridade dessa notícia (N_i) com a notícia da qual a lista de recomendação é criada (N_j).

$$SimRelev\_Index(N_i, N_j) = (\alpha * Jaccard(N_i, N_j)) + ((1-\alpha) * Cosine(N_i, V)) \quad (4.1)$$

Onde α é a constante que pondera o peso das medidas no índice; e V é o vetor global de pesos dos termos, obtidos através do algoritmo *TF-PDF*.

Na saída do Procedimento Recomendador, obtém-se, para cada notícia, uma lista de tamanho máximo K com as notícias relacionadas mais relevantes. Essa recomendação em forma de lista é armazenada em uma base de dados, pronta para ser retida e exibida através do *web-widget*.

5 EXPERIMETAÇÃO, AVALIAÇÃO E RESULTADOS

Para realizar os experimentos, foram coletadas, a partir do dia 20 de Janeiro de 2016, pouco mais de 5 mil notícias das páginas de 4 provedores de conteúdo: G1, UOL, Terra e R7. O processo de coleta foi efetuado por um *crawler* que percorre a *home page* e as páginas secundárias — páginas dos canais de esporte, economia, saúde, etc. — desses provedores de conteúdo em busca de *links* para notícias. As notícias coletadas distribuem-se distintamente em 145 canais.

As configurações do ambiente onde os experimentos foram realizados compreendem: um processador Intel® Core™i5-5200U; 8 GB de memória RAM; e Sistema Operacional Windows 10 Home 64 bits. Para recomendar 5 notícias relacionadas a cada uma das 1975 notícias coletadas na última semana (a partir do dia 20 de Janeiro de 2016), o algoritmo teve um tempo médio de execução de 74.52 segundos.

Como o algoritmo baseia-se em tendências de assuntos nos diversos canais para calcular a relevância dos termos, faz-se útil uma análise visual dessas tendências, bem como uma comparação com fatos da última semana (Figura 7). Para análise visual, optou-se por usar a técnica *nuvem de palavras*. Para cada canal, foram selecionados apenas alguns termos ordenados por relevância, para compor sua nuvem. Essa relevância é computada pelo índice *TF-PDF* e, na nuvem de palavras de cada canal, representa a expressividade do termo, i.e., quanto maior o índice *TF-PDF* do termo, maior ele aparecerá na nuvem de palavras. Os termos selecionados correspondem aos fatos relevantes, nacional e internacionalmente, da semana que compreende o período do dia 20 ao dia 27 de Janeiro de 2016. A nuvem *(b) Educação*, por exemplo, possui como termos mais expressivos: *"ensino público"*, *"MEC"*, *"Sisu"*, *"cursos"* e *"vagas"*, visto que o início de ano, normalmente, é o período de ingresso de muitos alunos em instituições de ensino superior através do ENEM e de vestibulares. Na nuvem *(d) Ciência e Saúde*, a palavra com maior expressividade é *"zika"*, já que o Brasil passa por uma situação delicada em relação as doenças transmitidas pelo mosquito *Aedes Aegypti*, que se agravam nessa período do ano.

Para comprovar a efetividade do algoritmo em recomendar notícias relacionadas, foram realizadas avaliações empíricas sobre as recomendações geradas (Figuras 8, 9, 10 e 11). O coeficiente de ponderação α, usado nas recomendações do experimento, foi 0.9. Isso significa que o índice de similaridade e relevância de uma notícia atribui 90% do seu valor à similaridade entre a notícia candidata e a notícia que receberá a recomendação, e 10% à relevância da notícia candidata. As avaliações demonstraram que o algoritmo tem a capacidade de gerar

recomendações sobre o mesmo tópico de uma notícia correntemente em leitura e ponderadas pela relevância de tópicos quentes. Assim, o Weews se põe como uma alternativa à seleção manual de recomendações, visto que essa última é propensa à falha humana e requer constantes intervenções para manter um tópico atualizado com notícias recentes.

Figura 7 – Nuvem de palavras para 4 canais de notícias

(a) Economia

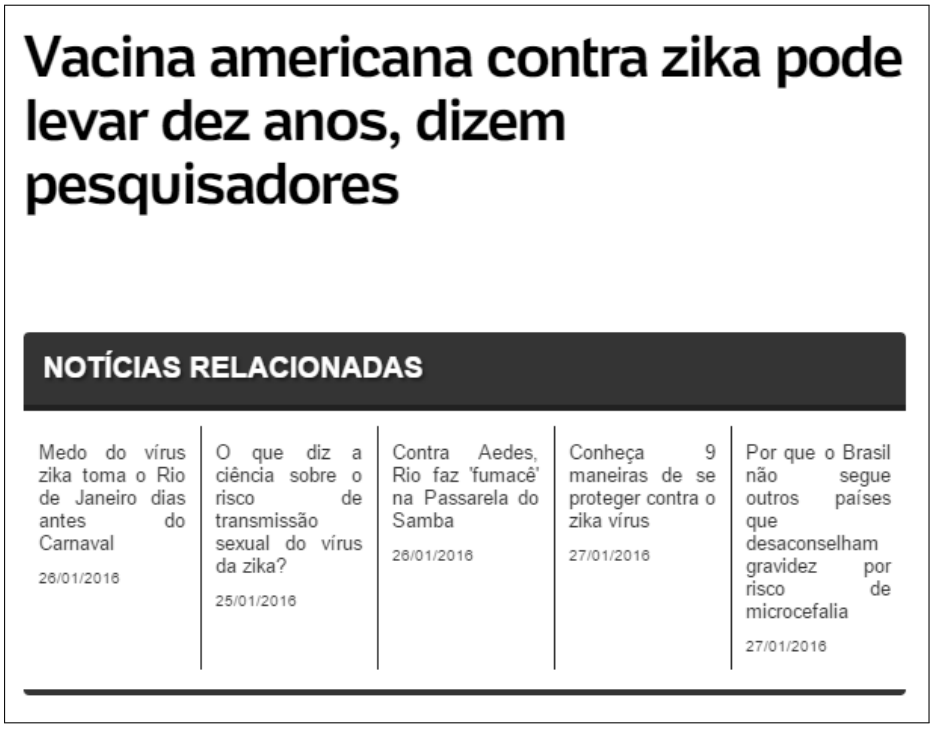
(b) Educação

(c) Oscar 2016

(d) Ciência e Saúde

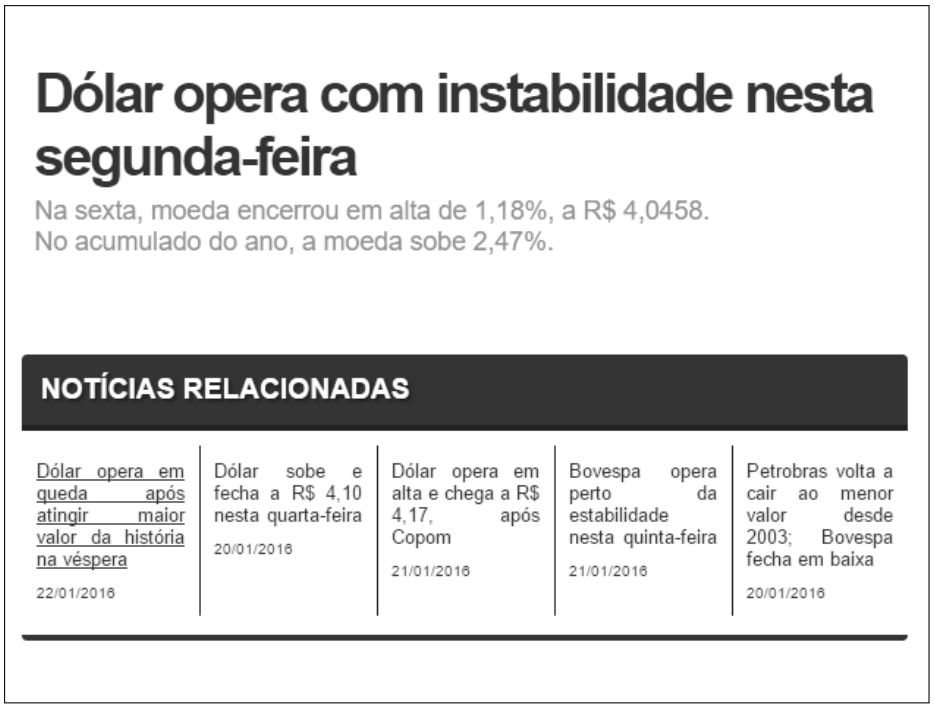
Fonte: autoria própria

Figura 8 – Exemplo de recomendação geradas pelo Weews para notícia do portal UOL



Fonte: Screeshot do UOL alterado digitalmente

Figura 9 – Exemplo de recomendação geradas pelo Weews para notícia do portal G1

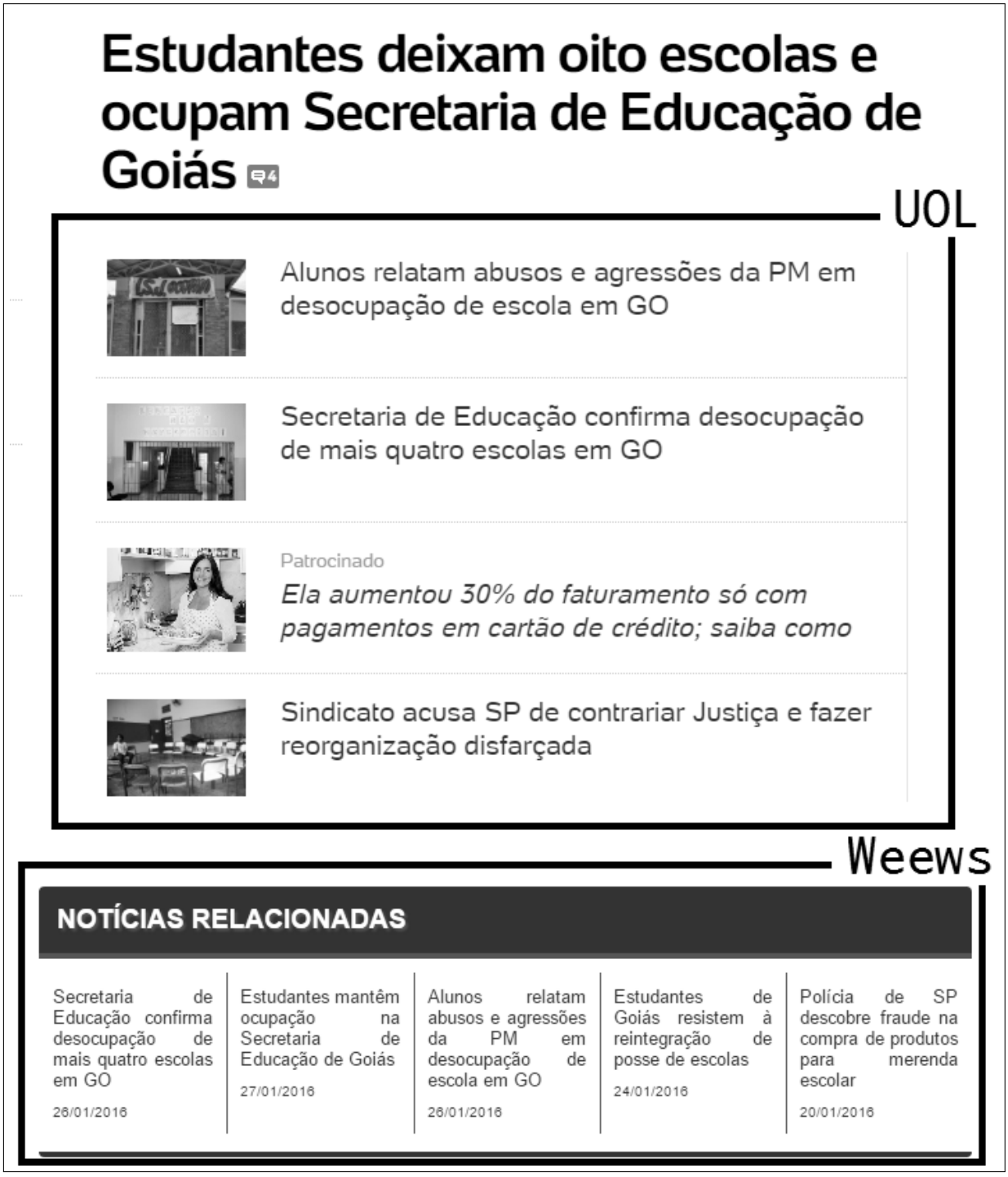


Fonte: Screeshot do portal G1 alterado digitalmente

Figura 10 – Exemplo de recomendação geradas pelo Weews para notícia do portal G1 (na parte inferior) comparada com as recomendações geradas pelo próprio G1 (na parte superior)

Fonte: Screeshot do portal G1 alterado digitalmente

Figura 11 – Exemplo de recomendação geradas pelo Weews para notícia do portal UOL (na parte inferior) comparada com as recomendações geradas pelo próprio UOL (na parte superior)



Fonte: Screeshot do portal UOL alterado digitalmente

6 CONCLUSÃO E TRABALHOS FUTUROS

Nesse trabalho, foi descrito o Weews, um sistema recomendador de notícias relacionadas que as apresenta em um *web widget* nas páginas das notícias de um portal. Para o Weews gerar as recomendações, foi desenvolvido um algoritmo de recomendação que usa filtragem baseada em conteúdo para selecionar k notícias relacionadas para cada notícia. Essa seleção se baseia no índice de similaridade e relevância entre as notícias, obtido de forma ponderada entre a similaridade Jaccard das entidades de duas notícias e a similaridade cosseno entre o texto de uma notícia candidata e o vetor de pesos dos termos do vocabulário. O vetor de pesos dos termos do vocabulário representa a relevância dos termos em um período t. Os termos mais relevantes são considerados *hot topics*, pois são termos frequentes em vários canais. Essa característica refina o resultado das recomendações e as mantém atualizadas. Foi também elaborada uma arquitetura que permite uma divisão de atribuições mais eficiente, permitindo respostas mais imediatas às requisições de consumo do serviço de recomendação do Weews.

Os resultados dos experimentos mostraram que o Weews se apresenta como uma alternativa eficiente às formas usadas atualmente para recomendar notícias relacionadas, visto que: (i) o algoritmo consegue atribuir altas relevâncias a termos que são ”quentes”na realidade e usa essa característica para ponderar as recomendações as mantendo atualizadas; e (ii) que as recomendações geradas relacionam o conteúdo das notícias de forma efetiva e automática.

O Weews pode ser estendido no sentido de tornar as recomendações personalizadas para os usuários. Mesmo que grande parte dos usuários acessem portais de notícias de anonimamente, ou seja, sem cadastro prévio, existem técnicas para contornar essa ausência de informações. Um exemplo é o uso de *cookies* gerados pelos portais e armazenados no *browser* do usuário. Esses *cookies* permitem distinguir usuários. Assim, um histórico de acesso pode ser obtido para montar listas personalizadas de recomendações com base no comportamento de leitura do usuário.

REFERÊNCIAS

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, IEEE, v. 17, n. 6, p. 734–749, 2005.

BOBADILLA, J. et al. Recommender systems survey. *Knowledge-Based Systems*, Elsevier, v. 46, p. 109–132, 2013.

BUN, K. K.; ISHIZUKA, M. Topic extraction from news archive using tf* pdf algorithm. In: IEEE. *null*. [S.l.], 2002. p. 73.

KOMPAN, M.; BIELIKOVÁ, M. Content-based news recommendation. In: *E-commerce and web technologies*. [S.l.]: Springer, 2010. p. 61–72.

LIU, J.; DOLAN, P.; PEDERSEN, E. R. Personalized news recommendation based on click behavior. In: ACM. *Proceedings of the 15th international conference on Intelligent user interfaces*. [S.l.], 2010. p. 31–40.

PHELAN, O.; MCCARTHY, K.; SMYTH, B. Using twitter to recommend real-time topical news. In: ACM. *Proceedings of the third ACM conference on Recommender systems*. [S.l.], 2009. p. 385–388.

RAJARAMAN, A. et al. *Mining of massive datasets*. [S.l.]: Cambridge University Press Cambridge, 2012. 73-130 p.

RICCI, F.; ROKACH, L.; SHAPIRA, B. *Introduction to recommender systems handbook*. [S.l.]: Springer, 2011.

ROCCHIO, J. J. Relevance feedback in information retrieval. Prentice-Hall, Englewood Cliffs NJ, 1971.

SCHAFER, J. B. et al. Collaborative filtering recommender systems. In: *The adaptive web*. [S.l.]: Springer, 2007. p. 291–324.

SCHEIN, A. I. et al. Methods and metrics for cold-start recommendations. In: ACM. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.], 2002. p. 253–260.