# Uma ferramenta para rastreamento semântico de eventos a partir de documentos da Web*

**Welton Santos , Leonardo Rocha**

[1] DCOMP/UFSJ - São João del-Rei, MG , Brasil

weltonsantos@aluno.ufsj.edu.br, lcrocha@ufsj.edu.br

***Abstract.*** *Exploring large news collections created by media outlets with traditional search engines is impractical for demanding users. Thus, we propose a temporal exploration tool that aims to facilitate the consultation of news collections. We concentrated our efforts on two fronts (i) allowing users to make queries with the addition of information from documents represented by word embbedings and (ii) developing a strategy for retrieving temporal information to generate timelines presented by an appropriate interface. We evaluated our solution in a collection of a Brazilian newspaper and demonstrated that it can draw different timelines, covering different subtopics of the same theme.*

***Resumo.*** *Explorar os grandes acervos criados por veículos de mídia com as ferramentas de busca tradicionais é algo impraticável para usuários exigentes. Assim, propomos uma ferramenta de exploração temporal que visa facilitar a realização de consultas em acervos de notícias. Concentramos nossos esforços em duas frentes (i) permitir que usuários façam consultas com adição de informações de documentos representados por word embbedings e (ii) desenvolver uma estratégia para resgate de informação temporal para gerar timelines apresentadas por uma interface adequada. Avaliamos nossa solução em um acervo de um jornal brasileiro e demonstramos que a mesma consegue traçar diferentes timelines, cobrindo diferentes subtópicos de um mesmo tema.*

## 1. Introdução

A evolução das máquinas de busca - *Searching Engine* - (e.g., Google, Bing, etc.) facilitou o acesso à informação. Essas ferramentas são baseadas em consultas por meio de palavras chaves, cujo objetivo é basicamente resgatar informação de uma base de dados a partir da associação da informação da base com poucas palavras chaves. Por exemplo, um economista que precisa encontrar notícias e informações sobre o mercado de ações irá informar algumas palavras chaves (e.g. mercado ações variações preço) para a máquina de busca e receberá as últimas notícias sobre eventos que ocasionaram oscilações no preços das ações (e.g., a notícia da fusão entre duas empresas) [Singh et al. 2016].

Apesar de se mostrarem eficientes para resgatar informações recentes e mais abrangentes, os buscadores tradicionais ainda possuem limitações para usuários que procuram temas mais específicos e temporalmente relacionados. Por exemplo, para um historiador interessado no comportamento do mercado de ações impactado por um evento influente (e.g. o impeachment da presidente Dilma Rousseff em 2016), exigirá do

*Esse trabalho foi parcialmente financiado por CNPq, CAPES, FINEP e Fapemig

historiador um trabalho oneroso e nem sempre praticável, forçando o mesmo a realizar várias consultas, ordenar e filtrar grandes volumes de dados. Dessa forma, para atender usuários mais exigentes, observamos que ainda existem dois desafios particulares durante o processo de pesquisa: (i) gerar boas consultas para resgatar documentos relevantes e mais específicos em base de dados e; (ii) ) organizar temporalmente grandes números de documentos (e.g., notícias, artigos) e explorá-los de forma eficiente [Alonso et al. 2009].

Dessa forma, o presente trabalho tem como objetivo oferecer uma ferramenta para usuários mais exigentes pesquisar, organizar e explorar documentos com facilidade a partir de *timelines* de documentos. Dado um *ranking* de documentos obtido por meio de uma máquina de busca a partir de uma *query*, o usuário escolhe o documento que esteja mais alinhado com seu interesse para que o mesmo seja utilizado como referência para a construção da *timeline*. Para construir a *timeline*, propomos um novo algoritmo, denominado *Timeline Builder* (TB), *nossa primeira contribuição*. Primeiramente, nosso algoritmo particiona os documentos retornados a partir da consulta do usuário em intervalos de tempo (e.g., semanas, quinzenas), de acordo com a data de criação dos documentos. Para cada intervalo, o TB define o conjunto de documentos que mais se assemelham à *query* original e ao documento de referência escolhido pelo usuário. Essa similaridade é calculada pelo TB utilizando as representações vetoriais dos documentos e da *query* por meio do modelo de *word embbedings* `doc2vec`. Para cada intervalo, o algoritmo substitui o documento de referência utilizado pelo documento mais similar a *query* e ao documento de referência do intervalo anterior para manter a correspondência temporal.

*Nossa segunda contribuição* consiste em uma ferramenta Web, que pode ser acoplada a qualquer máquina de busca, que organiza os intervalos de tempo gerados pelo TB em uma sequência de seções (*slides*), onde cada seção contém os documentos de um intervalo. Para auxiliar no processo de exploração, a ferramenta disponibiliza filtros de documentos por categorias (e.g., esportes, mercado) no *ranking* inicial e na construção da *timeline*. Além disso, permite também definir o número de documentos por intervalo gerando *timelines* mais sumarizadas e específicas.

Para avaliar nossa estratégia, simulamos uma máquina de busca sobre uma base de dados do jornal brasileiro Folha de São Paulo composta de 167 mil documentos. Nesta máquina de busca, o usuário insere uma *query* e seleciona um documento de referência entre os documentos gerados no *ranking* pela *query*. Como estudo de caso, utilizamos a *query "o impeachment de dilma rousseff e seus impactos"* e construímos diversas *timelines* para diferentes documentos de referência. Nossos resultados mostram que, a partir da variação do documento de referência, nosso algoritmo gera *timelines* que cobrem diferentes subtópicos (e.g., pedaladas fiscais, investigação Lava Jato). Além disso, avaliamos a importância do documento de referência em relação a *query* e percebemos que aumentando a importância do documento geramos *timelines* com documentos mais correlacionados à tópicos contidos dentro do documento de referência. Esse resultado é um forte indício de que o documento de referência funciona bem como uma fonte de informação. Até onde sabemos, não encontramos nenhum trabalho na literatura que visa realizar buscas por temas mais específicos e temporalmente relacionados.

## 2. Trabalhos Relacionados

### 2.1. Query Expansion

*Query expansion* é uma estratégia amplamente utilizada para enriquecer *queries* ambíguas ou com poucas palavras [Azad and Deepak 2019]. Muitos trabalhos focam em identificar novos termos para *queries* a partir da análise de um *ranking* de documento inicial [Rocchio 1971, Sparck Jones et al. 2000]. Após uma consulta inicial de documentos em uma máquina de busca, esta retorna um *ranking* com $K$ documentos que serão utilizados para enriquecer a mesma. Esta vertente é conhecida como *feature distribution of top ranked documents* FD-TRD, com destaque para as abordagens que utilizam Relevance Feedback (RF). Técnicas de RF *ranking* captam os interesses do usuário a partir de sua interação com os documentos do *ranking* [Rocchio 1971]. Trabalhos mais recentes exploram o potencial de modelos de *word embbedings* com RF. [Roy et al. 2016] apresentam uma abordagem baseada em KNN com *word embbedings* para encontrar novos termos associados a *query*. [Kuzi et al. 2016] mostram que técnicas com base em *word embbedings* podem superar técnicas tradicionais (e.g., RM3) utilizando RF.

### 2.2. Recuperação de Informação Temporal

Em [Kanhabua and Anand 2016] os autores fazem uma revisão sobre as técnicas de resgate de informação temporal e classificam os desafios existentes nessa vertente em: *temporal indexing and query processing, temporal query analysis e time-aware ranking*. Observando a literatura, percebemos que os trabalhos estão divididos em duas frentes: **(i)** desenvolver e aprimorar modelos temporais **(ii)** criar metáforas visuais para exploração de informação temporal. Seguindo a primeira frente [Alonso et al. 2009] desenvolvem um *add-on* para refazer rankings e gerar *timelines* com resultados de máquinas de busca. Com foco no agrupamento e sumarização de dados de redes sociais [Li and Cardie 2014] propõem uma técnica para gerar *timelines* de momentos importantes da vida de usuários no twitter. Para apresentação dos dados, [Matthews et al. 2010] apresentam a ferramenta *Time Explorer* para geração de *timelines* com associação dinâmica de entidades (e.g., celebridades, empresas). [Singh et al. 2016] apresentam uma ferramenta de maximização de cobertura de tópicos com foco em gerar *timelines* para historiadores em um interface amigável. Diferentemente dos trabalhos acima, este é o primeiro trabalho a criar *timelines* correlacionando documentos com auxílio de *word embbedings* e *Relevance Feedback*.

### 2.3. Representação de documentos

Modelos tradicionais baseados em Bag-Of-Words (BOW) possuem duas fraquezas inerentes as suas representações: **(i)** perda de informação semântica nas representações e **(ii)** os vetores gerados para representar documentos possuem grandes dimensões. Frente a estes problemas, modelos de *word embbedings* representam palavras em espaços vetoriais reduzidos preservando a semântica das palavras e documentos, com destaque aos trabalhos [Mikolov et al. 2013a, Mikolov et al. 2013b, Le and Mikolov 2014]. No trabalho [Mikolov et al. 2013a] os autores propõem o `word2vec`, um modelo amplamente utilizado. Basicamente, o trabalho propõe utilizar palavras em um espaço vetorial multidimensional, de modo que cada palavra seja representada por um vetor (geralmente de 50 a 2000 mil dimensões) em que palavras próximas semanticamente possuam vetores parecidos. Seguindo a linha de representação de palavras, [Le and Mikolov 2014] estendem a representação de palavras para representação de sentenças e documentos com o modelo amplamente utilizado `doc2vec`.

## 3. Story Tracker

Nessa seção detalhamos a *Story Tracker* (ST), nossa proposta para pesquisar, organizar e explorar documentos, apresentando os resultados por meio de *timelines*. Uma visão geral da ferramenta é apresentada na Figura 1 e as etapas serão detalhadas a seguir.

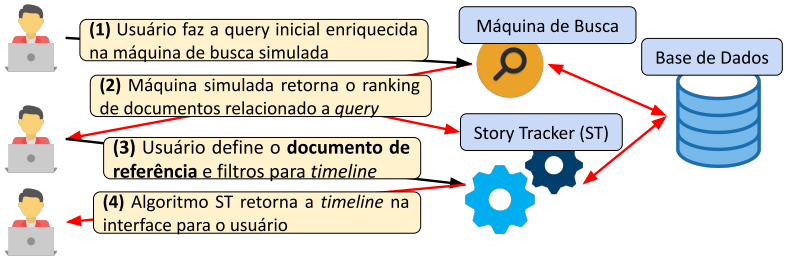


**Figura 1. Visão geral da interação do usuário na geração de *timelines*.**

### 3.1. Enriquecimento da Consulta

O primeiro componente da ST utiliza expansão de *queries* Relevance-Feedback RF para construir *timelines* sobre assuntos específicos com o auxílio da informação contida em documentos. O objetivo é permitir que o usuário crie *timelines* com base em assuntos específicos a partir de *queries* simples, explorando a informação de documentos como complemento. Para isso, anterior à construção da *timeline* a partir dos resultados de uma busca, mostramos ao usuário um *ranking* de documentos relacionados a uma *query*. Nesta fase o usuário fornece um *feedback* ao sistema sobre qual documento é mais relevante para ele. Assim, entre os documentos do *ranking*, o usuário escolhe um documento que seja mais de seu interesse e este documento será usado como fonte de informação para resgatar documentos junto a *query* na criação de *timelines* (seção 3.2). O processo de escolha do documento de referência pode ser visualizado na Figura 1 nas etapas 1 e 2.

Considerando uma máquina de busca que opera sobre uma base de documentos $D$ relativa a todos os documentos indexados e uma *query* $q$ criada pelo usuário com as palavras chaves sobre o tópico de seu interesse, construímos uma tupla $(q, C_q, p_q, m_q)$, onde $C_q$, $p_q$, $m_q$ também são definidos pelo usuário (etapa 1 Figura 1). $C_q$ é um conjunto de categorias utilizado para filtro onde somente documentos $(d_i \in D)$ com categorias em $C_q$ são recuperados $(Cat(d_i) \in C_q)$. $p_q$ é uma data base para exploração de documentos pela nossa solução. Por fim, $m_q$ é um período de tempo em meses que serve de raio para busca para data base $p_q$. A partir de $p_q$ e $m_q$ definimos o intervalo $T_q = [(p_q - m_q), (p_q + m_q)[$. Por exemplo, para $p_q = 2016/04/01$ e $m_q = 2$, documentos no período $T_q = 2016/02/01$ à $2016/06/01$ serão analisados. Assim, o escopo de documentos a serem analisados pelo *Timeline Builder* é dado por $E_q \in D$, tal que $E_q = \{d_i \in D | d_i \in T_q \wedge Cat(d_i) \in C_q\}$.

Dessa forma, a nova *query* $E_q$ é submetida à uma máquina de busca, a qual irá retornar para o usuário um conjunto de documentos $D_q$ mais relevantes. $D_q$ é enviada para o usuário, o qual irá escolher um documento de referência (etapa 2 Figura 1) para ser utilizado com fonte de informação no processo de construção de *timeline* descrito na seção 3.2. Para cada um dos documentos de $D_q$, o *Timeline Builder* irá criar uma representação vetorial utilizando o modelo *word embbedings* `doc2vec`. Da mesma forma, a consulta $q$ também será transformada em uma representação vetorial a partir da combinação linear de todos do vetores de cada palavra de $q$. É a partir do cálculo de similaridade dessas representações vetoriais de documentos e *query* que a *timeline* será criada.

### 3.2. Construção da Timeline

O foco do algoritmo *Timeline Builder* (TB) reside na exploração da similaridade entre documentos pertencentes ao resultado de uma consulta específica, publicados em períodos de tempo próximos (e.g., mesma semana), uma vez que estes tendem a ser mais correlacionados. Desta forma, o cerne do algoritmo aqui proposto consiste em relacionar documentos temporalmente, de modo que, a partir de uma *query* e um documento de referência, documentos publicados em intervalos de tempo vizinhos sejam resgatados e virem fonte de informação para recuperação de outros documentos. A aplicação do algoritmo TB sobre a base de dados é referente às etapas 3 e 4 do exemplo da Figura 1.

O algoritmo TB recebe uma tupla de parâmetros $(q, d_r, D_q, T_q, s)$. $d_r$ é o documento de referência escolhido pelo usuário, $s$ refere-se à granularidade em dias entre os intervalos que compõem a *timeline* e $q, D_q$ e $T_q$ são definidos como anteriormente. O algoritmo TB divide a construção da *timeline* em duas partes: passado e futuro. A partir da data de publicação do documento de referência $d_r$ ($Pub(d_r)$) o algoritmo cria dois conjuntos de intervalos (subespaços de tempo) $T_{pas}$ e $T_{fut}$ referentes ao passado e ao futuro de $Pub(d_r)$ respectivamente. Estes conjuntos de intervalos, por sua vez, consistem na divisão do período $T_q$ em $T$ intervalos com tamanho $s$ em dias. Para gerar a *timeline*, é aplicado o processo de expansão descrito abaixo nos conjuntos de intervalos $T_{pas}$ e $T_{fut}$ de forma similar. Após a expansão de cada conjunto de intervalos as saídas geradas para o passado e futura são concatenadas e apresentadas para o usuário como uma *timeline*.

#### 3.2.1. Expansão

Para cada intervalo de tempo $t \in T$, o TB cria subconjuntos de documentos $S_t$ (Algoritmo 1 linhas 3-6) com os documentos $d_t \in D_q$ e cuja data de publicação esteja dentro do intervalo $t$. Para cada $d_t$ é calculada sua importância (proximidade semântica) em relação à *query* e ao documento de referência $d_r$ (linha 5), por meio da função $util(q, d_r, d_i) = \alpha.cos(q, d_t)+(1-\alpha).cos(d_r, d_t)$. Esta função equilibra a importância entre *query* $q$ e $d_r$ em relação a cada $d_t$ através do parâmetro $\alpha$, que varia no intervalo [0,1], de modo que, quanto maior $\alpha$, maior a importância da *query* e vice-versa. O parâmetro $\alpha$ controla também a quantidade de ruído adicionado no uso do documento de referência, uma vez que, $d_r$ pode abordar diversos assuntos, todos relacionados a um assunto principal. Com valores mais altos de $\alpha$ o usuário pode reduzir o impacto de possível informação ruidosa e deixar os resultados mais próximos da *query* original. O $S_t$ é então ordenado de forma decrescente de acordo com o valor da função $util$ calculada para cada um de seus documentos (linha 10) e esse conjunto é concatenado a $S$, o conjunto de todos os documentos $D_q$ ordenados de forma cronológica e pela sua relevância para a *timeline* criada.

**Algoritmo 1:** TIMELINE BUILDER $(q, d_r, D_q, T_q, s)$

```
1  T ← T_q / s;
2  foreach interval t ∈ T do
3      S_t ← {};
4      foreach document d_t where ((d_t ∈ D_q)&(Pub(d_t) ∈ t)) do
5          d_t.util ← util(q, d_r, d_t);
6          S_t ← S_t ∪ d_t
7      sort_util(S_t, descending);
8      d_r ← MAX(S_t);
9      S ← S ∪ S_t;
10 return S
```

### 3.2.2. Atualização do documento de referência

Com base no pressuposto que documentos de um intervalo tendem a ser mais similares aos documentos de seus intervalos vizinhos, para buscar documentos mais similares, o algoritmo TB atualiza o documento de referência a cada subconjunto de documentos de cada intervalo. Partindo do intervalo de tempo $t_0$ o algoritmo TB recupera os documentos mais relacionados a *query* $q$ e ao documento de referência escolhido pelo usuário, denominado $d_{r0}$. Após a construção do subconjunto de documentos do intervalo $t_0$ o algoritmo avança para o intervalo $t_1$. Na construção do subconjuntos de documentos deste intervalo o documento de referência $d_{r0}$ é substituído por outro documento denominado $d_{r1}$ (linha 8) através da função $MAX$, a qual retorna o documento de maior relevância (função $util$) dentro do tempo $t_0$. Generalizando, podemos definir o documento de referência utilizado no intervalo $t_n$ como o mais relevante no intervalo $t_{n-1}$.

## 3.3. Interface Gráfica

Conforme a Figura 1 a construção da *timeline* é realizada a partir de duas etapas desacopláveis entre si: enriquecimento de *query* e a construção de *timeline*. O enriquecimento da *query* consiste no primeiro contato do usuário com a ferramenta e nesta etapa o usuário interage com a interface apresentada na Figura 2. Nesta etapa, o usuário enriquece a *query* configurando a data base e a cobertura da *timeline* nas caixas dois e três respectivamente. Na Figura 2 podemos ver um exemplo de *ranking* inicial com os *cards* contendo o título do documento e uma caixa de seleção para marcar o documento a ser utilizado como documento de referência (caixa 5). Na caixa 4 o usuário pode definir a quantidade de documentos no *ranking* inicial. Além dos parâmetros presentes na interface inicial, o usuário pode enriquecer a sua *query* com parâmetros avançados, acionados no botão "*Advanced Search*" como destacado em verde na Figura 2. Nesta guia são configurados o conjunto de categorias para filtros de documentos (caixa 6), a granularidade dos intervalos de tempo (caixa 7) e o balanço entre a importância da *query* e o documento de referência (caixa 8).

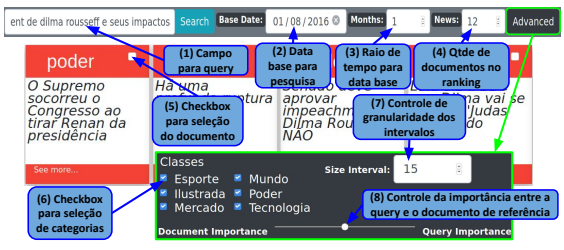


**Figura 2. Tela de construção do *ranking* inicial de documentos.**

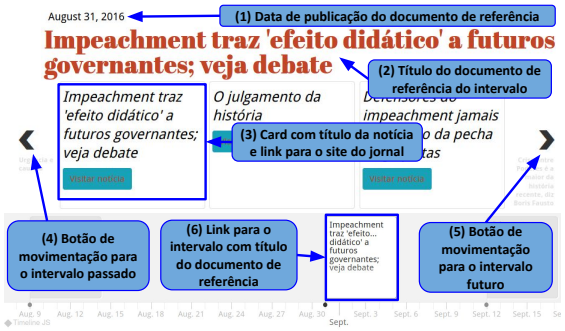


**Figura 3. Tela com documentos de um intervalo da *timeline*.**

Após selecionar um documento de referência e considerando todos os documentos retornados pela *query*, o algoritimo TB constroí a *timeline* e a retorna para o usuário conforme interface presente na Figura 3. Nesta tela, são apresentados ao usuário os documentos mais associados a *query* e ao documento de referência no intervalo. Destacado em azul nas caixas 1 e 2 estão o título e da data do documento de referência, respectivamente. Nas caixas 3 e 4 estão destacados os botões utilizados pelo usuário para navegar pelos intervalos de tempo da *timeline*. Sinalizado pela caixa 6, a interface oferece outra forma de movimentar pela *timeline* através da barra inferior. Cada link na barra contém o título do documento de referência do itervalo e uma função de redirecionamento para o mesmo.

## 4. Avaliação Experimental

### 4.1. Base de Dados

Modelos de *word embbedings* são famosos por capturar informação semântica entre palavras e documentos [Mikolov et al. 2013a, Le and Mikolov 2014]. Neste trabalho, exploramos a capacidade destes modelos, mais especificamente do modelo `doc2vec`[1] para encontrar documentos mais fortemente correlacionados entre si. Para treinar nossos modelos, utilizamos a base de documentos do Jornal Folha de São Paulo[2] que contem mais de 167 mil documentos, no período de Janeiro de 2015 a Setembro de 2017. Cada documento possui o título da matéria, conteúdo textual, data de publicação, categoria e link para página. Para cada documento, extraímos o conteúdo de texto do mesmo e removemos palavras sem relevância (i.e. *stop words*). A partir dessa extração, treinamos um modelo de representação `doc2vec` por meio do algoritmo *Paragraph-Vector Distributed Memory* PV-DM, considerando o seguinte conjunto de parâmetros do algoritmo (100 dimensões, 10 épocas, etc.). Devido ao grande volume de categorias redundantes entre os documentos da base, utilizamos um modelo de regressão logística[3] com as representações vetorias do modelo PV-DBOW *Paragraph-Vector Distributed Bag-of-Words* e redistribuímos os documentos em seis principais categorias (esporte, ilustrada, mercado, mundo, poder e tecnologia).

### 4.2. Máquina de Busca

Para avaliar nossa estratégia de uma maneira consistente e controlada, simulamos uma máquina de busca sobre a base de dados descrita anterioriormente. A máquina opera sobre um subconjunto de documentos da base de dados definido pelo escopo de uma *query* enriquecida do usuário (seção 3.1) e apresenta para o usuário o *ranking* dos $K$ documentos mais próximos a *query*. Para estimar a semelhança entre os documentos e a *query* $q$, primeiro criamos a representação vetorial $v_q$ de $q$ como a combinação linear dos vetores das palavra de $q$. O vetor de cada palavra é extraído do modelo PV-DM. Palavras não existentes no modelo são ignoradas no cálculo. Com o vetor $v_q$ estipulado, medimos a similaridade de cosseno de $v_q$ com todos os documentos no escopo da *query* definida pelo usuário e retornamos o conjunto de $K$ documentos mais semelhantes a *query*.

### 4.3. Resultados

Nesta seção discutimos o impacto da ferramenta aqui proposta na construção de *timelines*. Analisamos a capacidade da nossa proposta em construir diferentes *timelines* correlacionando documentos a partir de uma única *query* de entrada, variando o documento de referência e o limiar $\alpha$. Para construção de *timelines*, utilizamos como caso de estudo o processo de *impeachment* da ex-presidente do Brasil, Dilma Rousseff. Escolhemos este tópico porque o processo de *impeachment* durou um longo período de tempo, gerando grandes volumes de notícias. Na entrada de nossa máquina de busca utilizamos a *query "o impeachment de dilma rousseff e seus impactos"*, com data base $2016/08/01$ (um mês antes a data do *impeachment*), raio de tempo de um mês (total de dois meses um futuro e outro passado) e a granularidade dos intervalos de 15 dias. Como

[1] `https://radimrehurek.com/gensim/models/doc2vec.html`
[2] `https://www.kaggle.com/marlesson/news-of-the-site-folhauol`
[3] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`


**Figura 4. *Ranking* com seis documentos mais próximos a query "o impeachment de dilma rousseff e seus impactos".**


**Figura 5. Matriz com dupla de documentos mais relevantes no intervalo pelo documento de referência e limiar.**

conjunto de categorias consideramos *poder, mercado* e *mundo*. Sobre estes parâmetros a máquina de busca retornou o *ranking* de documentos contido na Figura 4.

Dado o *ranking* de notícias (Figura 4) tomamos para análise os documentos três (D3)(*"Senado deve aprovar impeachment de Dilma Rousseff? NÃO"*) e seis (D6)(*"Presidente do BC diz que interesses particulares atrapalham ajuste fiscal"*). Escolhemos estes documentos devido a divergência no foco principal entre estes, política e economia, respectivamente. Assim, ilustramos a aplicação da técnica de RF simulando a contribuição de usuário enviando *feedbacks* para o sistema sobre quais documentos são relevantes, como descrito na seção 3.1. Como base da discussão deste trabalho geramos quatro *timelines* TM1 e TM2 com $\alpha = 0.5$ para os documentos D3 e D6, respectivamente e TM3 e TM4 para o documento D6 com $\alpha = \{0.2, 0.5\}$. Para análise selecionamos os dois principais documentos de três intervalos das *timelines* como mostrado na Figura 5.

Observando na Figura 5 os documentos em TM1 obtidos a partir da *query* mais o documento de referência D3, notamos que todos os principais documentos dos três intervalos tratam do tema *impeachment* com tópico principal. Dentre os tópicos pautados pelos documentos no geral, estes estão associados à temas como "fraudes fiscais", "investigações de corrupção" e "golpe parlamentar". No geral, estes temas compartilham como foco principal o processo de *impeachment* tangenciando outros tópicos. Como exemplo, podemos citar o terceiro (*"Impeachment traz 'efeito didático' a futuros governantes; veja debate"*) e quarto (*"O julgamento da história"*) documentos que abordam o evento conhecido como "pedaladas fiscais". Comparando os documentos em TM2 com os documentos em TM1, é perceptível que documentos mais ligados à economia surgem em meio aos demais. Como principais exemplos dessa alteração mostram-se o segundo (*"Pré-delação da Odebrecht faz dólar avançar; Bolsa sobe com Petrobras"*) e terceiro (*"Destino de reformas propostas por Temer preocupa investidores"*) documentos em TM2 que descorrem sobre os tópicos "variação no preço de ações" e "preocupação de investidores extrangeiros", respectivamente. Ambos documentos tratam de tópicos que utilizam o *impeachment* como plano de fundo devido a instabilidade econômica que este provocou no país. Com base nesta análise, podemos perceber o impacto causado com a alteração do documento de referência, uma vez que com $\alpha = 0.5$ já foi possível direcionar a construção da *timeline* a partir, somente, da alteração de D3 para D6.

Como já descrito, a segunda forma de se construir *timelines* diferentes é a partir da variação do parâmetro $\alpha$ para a mesma tupla de *query* e documento de referência. Desta forma, quanto maior $\alpha$ maior a similaridade dos documentos à *query* e vice-versa. Para analisar o impacto deste parâmetro, utilizamos o documento D6 para gerar as *timelines* T3 e T4 com $\alpha = \{0.2, 0.8\}$. Escolhemos este documento para observarmos o quanto a variação do limiar $\alpha$ afasta e aproxima os resultados de tópicos sobre economia.

Como mostrado na Figura 5, variando $\alpha$ de 0.5 (TM2) para 0.8 (TM4) apenas o quarto documento em T4 (*"Reajuste para ministros do STF causa divergência entre PSDB e PMDB"*) relaciona tópicos de economia. Embora este documento esteja levemente relacionado à economia por conter diversos termos de cunho econômico (e.g., salário, desemprego, endividamento), lendo este documento, notamos que este possui foco nas discussões entre partidos políticos, ocasionadas por causa de ajustes salariais não apropriados. Por outro lado, variando $\alpha$ de 0.5 (TM2) para 0.2 (TM3), podemos perceber que todos os documentos principais dos intervalos possuem cunho econômico como foco principal, no geral, abordando tópicos como "preço de ações", "inflação" e "crescimento do PIB". Como exemplo visível do favorecimento a tópicos de economia tangentes ao documento D6 ocasionados pela redução do limiar $\alpha$, podemos utilizar o terceiro (*"Pré-delação da Odebrecht faz dólar avançar; Bolsa sobe com Petrobras"*) documento em TM2 que tornou-se o documento mais importante de seu intervalo em TM3. Este documento aborda as oscilações no mercado de ações ocasionados por investigações de crimes da empresa Odebrecht[4] e sua associação com a petrolífera Petrobras[5]. Essa característica justifica o aumento de sua relavancia dada a redução do limiar $\alpha$.

Por final, comparando os principais documentos em TM3 e TM4, podemos perceber que ambas as *timelines* não compartilham nenhum documento principal entre si, apesar de abordarem tópicos que são constantemente relacionados. Observando o último documento em TM3, documento de cunho econômico, observamos que este não faz menção direta ao processo de *impeachment*. Porém, percebemos que, este ainda traz sentenças relacionadas ao *impeachment*, como "O Congresso pode estar um tumulto, fugindo da polícia". Este comportamento evidencia o sucesso na captura de informação semântica entre documentos por nossos modelos, permitindo que nossa metodologia relacione documentos tangentes a partir da informação implicitamente compartilhada. Nos mostra também, a eficiência do limiar $\alpha$ no direcionamento da construção de *timelines*.

## 5. Conclusão e Trabalhos Futuros

Apresentamos neste trabalho o *StoryTracker*, uma ferramenta para organizar e explorar documentos oriundos de uma busca com facilidade a partir de *timelines*. Nossa proposta é composta de três partes principais: (1) Enriquecimento de Consulta; (2) Construção da *timeline*; e (3) Interface Gráfica. Na primeira parte permitimos que o usuário enriqueça sua consulta adicionando junto à sua *query* outros parâmetros (e.g., categoria de documentos), além de selecionar um documento de referência como fonte de informação para construção da *timeline*. Na segunda parte o algoritmo ST identifica os documentos mais similares à consulta e ao documento de referência, estabelecendo-se assim uma *timeline*. A terceira etapa consiste de um conjunto de metáforas visuais intuitivas as quais permitem ao usuário explorar de forma organizada os dados com praticidade.

[4] `https://en.wikipedia.org/wiki/Odebrecht`
[5] `https://en.wikipedia.org/wiki/Petrobras`

Para avaliar o *StoryTracker*, simulamos uma máquina de busca utilizando uma base de artigos publicados pelo jornal brasileiro Folha de São Paulo, composta de 167 mil documentos. Simulando uma busca pelos termos "o impeachment de dilma rousseff e seus impactos", construímos diferentes *timelines*, avaliando duas perspectivas distintas: política e econômica. Nossos resultados mostram que a partir da variação do documento de referência, a ferramenta proposta consegue traçar diferentes *timelines*, cobrindo diferentes subtópicos do processo de impeachment (e.g., investigação Lava Jato, pedaladas fiscais), os quais não seriam possíveis por meio de uma única *query* em uma máquina de busca tradicional. Como trabalhos futuros, nossa meta é acoplar o *StoryTracker* a diferentes máquinas de busca atuais, realizando uma avaliação online dessa combinação sob a perspectiva de usabilidade, considerando diferentes perfis de usuários.

## Referências


Alonso, O., Gertz, M., and Baeza-Yates, R. (2009). Clustering and exploring search results using timeline constructions. In *Proceedings of ACM CIKM*.

Azad, H. K. and Deepak, A. (2019). Query expansion techniques for information retrieval: A survey. *Information Processing and Management*, 56(5):1698–1735.

Kanhabua, N. and Anand, A. (2016). Temporal information retrieval. In *Proceedings of ACM SIGIR*.

Kuzi, S., Shtok, A., and Kurland, O. (2016). Query expansion using word embeddings. In *Proceedings of ACM CIKM*.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of ICML*.

Li, J. and Cardie, C. (2014). Timeline generation: Tracking individuals on twitter. In *Proceedings of ACM WWW*.

Matthews, M., Tolchinsky, P., Blanco, R., Atserias, J., Mika, P., and Zaragoza, H. (2010). Searching through time in the new york times. In *Proceedings of ACM HCIR*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. In *Proceedings of ICLR*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Proceedings of ICNIPS*.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. In *Proceedings of The Smart retrieval system - experiments in automatic document processing*.

Roy, D., Paul, D., Mitra, M., and Garain, U. (2016). Using word embeddings for automatic query expansion. *ArXiv*, abs/1606.07608.

Singh, J., Nejdl, W., and Anand, A. (2016). History by diversity: Helping historians search news archives. In *Proceedings of ACM CHIIR*.

Sparck Jones, K., Walker, S., and Robertson, S. (2000). A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Information Processing & Management*, 36(6):779–808.