Ovídio José Francisco

Aplicação de técnicas de Recuperação de Informação para Organização e Extração de Históricos de Decisões de Documentos de Reuniões

Sorocaba, SP

7 de janeiro de 2018

Sumário

1 Introdução

A popularização dos computadores possibilitou o armazenamento cada vez maior de conteúdos digitais, sendo bastante comum, o formato textual como livros, documentos, e-mails, redes sociais e páginas web. A produção de textos gera fontes de informações em volumes crescentes que podem superar a capacidade humana de analisá-los manualmente. Essa dificuldade incentiva a pesquisa de ferramentas automáticas para manipulação de dados não estruturados. Assim, os processos de extração automática de conhecimento em coleções textuais são essenciais, e ao mesmo tempo, constituem um desafio, devido às características de documentos textuais como o formato não estruturado e trechos com diferentes níveis e importância, desde informações essenciais até textos pouco informativos e em alguns casos até irrelevantes.

Além dos tipos de informações mais comuns que são armazenados no formato textual, como e-mails, relatórios, artigos e postagens em redes sociais, têm-se também o armazenamento das atas de reuniões, as quais permitem às organizações a documentação oficial de reuniões em arquivos digitais, facilitando a confecção, compartilhamento e consulta às decisões tomadas. Reuniões são tarefas presentes em atividades corporativas, ambientes de gestão e organizações de um modo geral, onde discute-se problemas, soluções, propostas, alterações de projetos e frequentemente são tomadas decisões importantes onde a comunicação entre os membros da reunião é feita de forma majoritariamente verbal. Para que seu conteúdo possa ser registrado e externalizado, adota-se a prática de escrever seu conteúdo em documentos, chamados atas.

Por exemplo, nas reuniões do conselho de um programa de pós-graduação de uma universidade, são decididos, quais são os critérios para credenciamento e permanência de docentes no programa. Ao longo do tempo, esse tema pode ser discutido e mencionado diversas vezes, podendo os critérios inclusive passar por significativas alterações, devido a diversos fatores. O coordenador do programa pode desejar recuperar qual foi a decisão mais recente, para poder aplicar os critérios a um potencial novo membro do programa, ou os membros do conselho podem desejar rever o histórico de tudo o que já foi discutido/decidido sobre o tema, para poder propor alterações nas regras, de forma mais adequada.

As atas de reunião possuem características particulares. Frequentemente apresentam um texto com poucas quebras de parágrafo e sem marcações de estrutura, como capítulos, seções ou quaisquer indicações sobre o tema do texto. Devido a fatores como a não estruturação e volume dos textos, a localização de um assunto em uma coleção de atas é uma tarefa custosa, especialmente considerando o seu crescimento em uma instituição. As organizações costumam manter seus documentos eletrônicos organizados em pastas e

nomeá-los com informações básicas sobre a reunião a que se refere como a data e alguma referência cronológica, por exemplo "37ª Reunião Ordinária do Conselho ...". Essa forma de organização facilita a localização dos arquivos com ferramentas que fazem buscas pelo nome dos arquivos e pastas. Contudo, essa prática costuma ser insuficiente, pois uma busca pelo conteúdo dos textos usa-se ferramentas computacionais baseadas em localização de palavras-chave que além de encontrar ocorrências das palavras podem oferecer recursos como operadores *and* e *not* ou ainda suporte às expressões regulares. Esse recurso, conhecido como *grepping*[1], traz resultados satisfatórios em muitos casos. Por outro lado, traz algumas desvantagens como: 1) transfere certa complexidade da tarefa ao usuário 2) as buscas em grandes coleções de documentos podem ser mais lentas 3) não há suporte a padrões mais flexíveis como a proximidade entre as palavras ou palavras que estejam na mesma sentença 4) o retorno ao usuário são os documentos integrais, o que pode exigir uma segunda busca dentro de um documento para encontrar o trecho desejado.

Para superar essas limitações têm sido utilizadas técnicas de aprendizado de máquina por meio de diversas abordagens. Por exemplo, elas vêm sendo empregadas na organização, gerenciamento, recuperação de informação e extração de conhecimento, como a extração de tópicos e a categorização de automática de documentos. Essas técnicas permitem melhorar a busca por informações em atas de reunião.

Uma vez que a ata registra a sucessão de assuntos discutidos na reunião, um sistema de recuperação de informação idealmente deve retornar ao usuário apenas o trecho que trate do assunto pesquisado ao invés do documento inteiro. Assim, cada trecho com um assunto predominate pode ser considerado um subdocumento. Portanto, em primeiro lugar, há a necessidade de descobrir onde há mudanças de assunto no texto. Técnicas de segmentação automática de textos (segmentação textual) podem ser aplicadas com esse propósito.

A tarefa de segmentação automática de textos, ou segmentação textual consiste em dividir um texto em partes que contenham um significado relativamente independente. Em outras palavras, é identificar as posições nas quais há uma mudança significativa de assunto. É útil em aplicações que trabalham com textos sem indicações de quebras de assunto, ou seja, não apresentam seções ou capítulos, como transcrições automáticas de áudio, vídeos e grandes documentos que contêm vários assuntos como atas de reunião e notícias.

Pode ser usada para melhorar o acesso a informação solicitada por meio de uma consulta, onde é possível oferecer porções menores de texto mais relevantes ao invés de exibir um documento grande que pode conter informações menos pertinentes. Além disso, encontrar pontos onde o texto muda de assunto, pode ser útil como etapa de pré-processamento em aplicações voltadas ao entendimento do texto, principalmente em

[1] O nome *grepping* é uma referência ao comando `grep` do Unix

documentos longos (CHOI, 2000).

Assim, nesse contexto, este trabalho propõe a investigação do uso de mineração de texto e as técnicas que constituem o estado da arte na área para o desenvolvimento de uma ferramenta para extração automática de históricos de decisão em atas de reuniões.

2 Conceituação Teórica

A popularidade dos computares permite a criação e compartilhamento de textos onde a quantidade de informação facilmente extrapola a capacidade de humana de leitura e análise de coleções de documentos, estejam eles disponíveis na Internet ou em computadores pessoais. A necessidade de simplificar e organizar grandes coleções de documentos criou uma demanda por modelos de aprendizado de máquina para extração de conhecimento em bases textuais. Para esse fim, foram desenvolvidas técnicas para descobrir, extrair e agrupar textos de grandes coleções, entre essas, a modelagem de tópicos (HOFMANN, 1999; DEERWESTER et al., 1990; LEE; SEUNG, 1999; BLEI, 2012).

2.1 Segmentação Textual

A tarefa de segmentação textual consiste em dividir um texto em partes ou segmentos que contenham um significado relativamente independente. Em outras palavras, é identificar as posições nas quais há uma mudança significativa de assuntos. As técnicas de segmentação textual consideram um texto como uma sequência linear de unidades de informação que podem ser, por exemplo, cada termo presente no texto, os parágrafos ou as sentenças. Cada unidade de informação é um elemento do texto que não será dividido no processo de segmentação e cada ponto entre duas unidades é considerado um candidato a limite entre segmentos. Nesse sentido, um segmento pode ser visto como uma sucessão de unidades de informação que compartilham o mesmo assunto.

Para encontrar os segmentos de um texto, alguns dos primeiros algoritmos utilizam a técnica de janelas deslizantes, onde se verifica a frequência dos termos em um fragmento do documento. Inicialmente, estabelece-se a partir do início do texto, um *range* de w termos, chamado janela que em seguida é deslocada em passos de k termos adiante até o final do texto. A cada passo, analisa-se os termos contidos na janela.

Trabalhos anteriores se apoiam na ideia de que a mudança de assunto em um texto é acompanhada de uma proporcional mudança de vocabulário. Essa ideia, chamada de coesão léxica, sugere que a distribuição das palavras é um forte indicador da estrutura do texto (KOZIMA, 1993). O autor demonstrou que há uma estreita correlação entre quedas na coesão léxica em janelas de texto e a transição de assuntos. Em seu trabalho, calculou a coesão léxica de uma janela de palavras usando *spreading activation* em uma rede semântica especialmente elaborada para o idioma Inglês. Contudo, a implementação de um algoritmo para outros domínios dependia da construção de uma rede adequada.

A partir desses conceitos, um dos primeiros algoritmos baseados na ideia que um

segmento pode ser identificado pela análise das palavras que o compõe foi o *TextTiling*. O *TextTiling* é um algoritmo baseado em janelas deslizantes, em que, para cada candidato a limite, analisa-se o texto circundante. O *TextTiling* recebe uma lista de candidatos a limite, usualmente finais de parágrafo ou finais de sentenças. Para cada posição candidata são construídos 2 blocos, um contendo sentenças que a precedem e outro com as que a sucedem. O tamanho desses blocos é um parâmetro a ser fornecido ao algoritmo e determina o tamanho mínimo de um segmento. Esse processo é ilustrado na Figura 1.

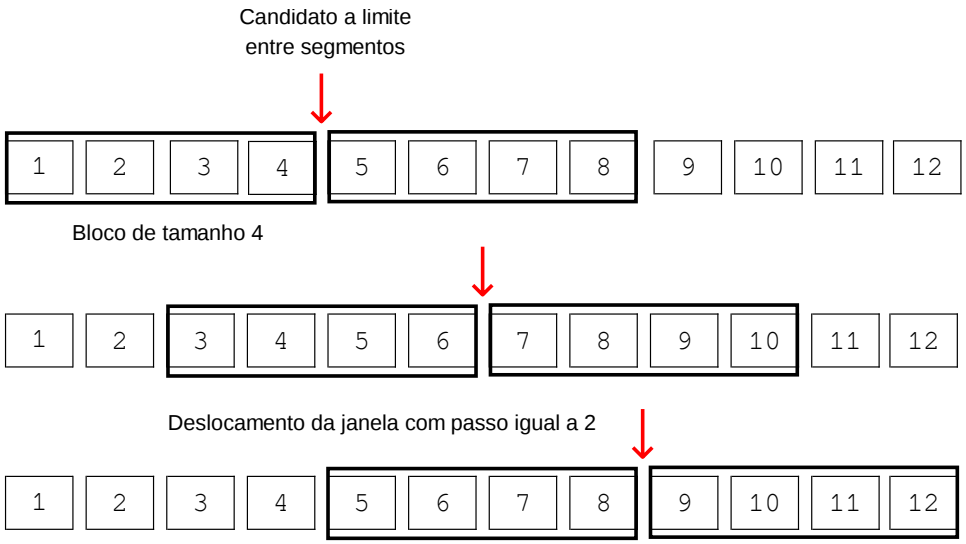

Figura 1 – Processo de deslocamento da janela deslizante. Os quadrados numerados representam as sentenças e os retângulos representam os blocos de texto a serem comparados. O deslocamento movimenta o candidato a limite e por consequência os blocos que o antecede e precede.

Em seguida, os blocos de texto são representados por vetores que contém as frequências de suas palavras. Diferente da proposta de Kosima, utiliza *cosine* como medida para a similaridade entre os blocos adjacentes, conforme apresentada na Equação 2.1, onde dados dois blocos de texto, x e y, $f_{x,j}$ é a frequência do termo j em x e $f_{y,j}$ é a frequência do termo j em y.

$$Sim(x,y) = \frac{\Sigma_j f_{x,j} \times f_{y,j}}{\sqrt{\Sigma_j f_{x,j}^2 \times \Sigma f_{y,j}^2}} \tag{2.1}$$

Um limite ou transição entre segmentos é identificado sempre que a similaridade entre as unidades que antecedem e precedem o ponto candidato cai abaixo de um limiar, indicando uma diminuição da similaridade entre os blocos adjacentes. Ou seja, identifica-se uma transição entre segmentos pelos vales na curva de dissimilaridades. Para cada final de sentença representada por y_i atribui-se uma profundidade dada por $(y_{i-1} - y_i) + (y_{i+1} - y_i)$ e será um limite entre segmentos caso a profundidade exceda $\overline{s} - \sigma$, onde $\overline{s}$ é a média da profundidade de todos os vales do documento e σ, o desvio padrão. Na Figura 2 é

ilustrado os deslocamentos da janela deslizante e a curva de dissimilaridade entre os blocos adjacentes.

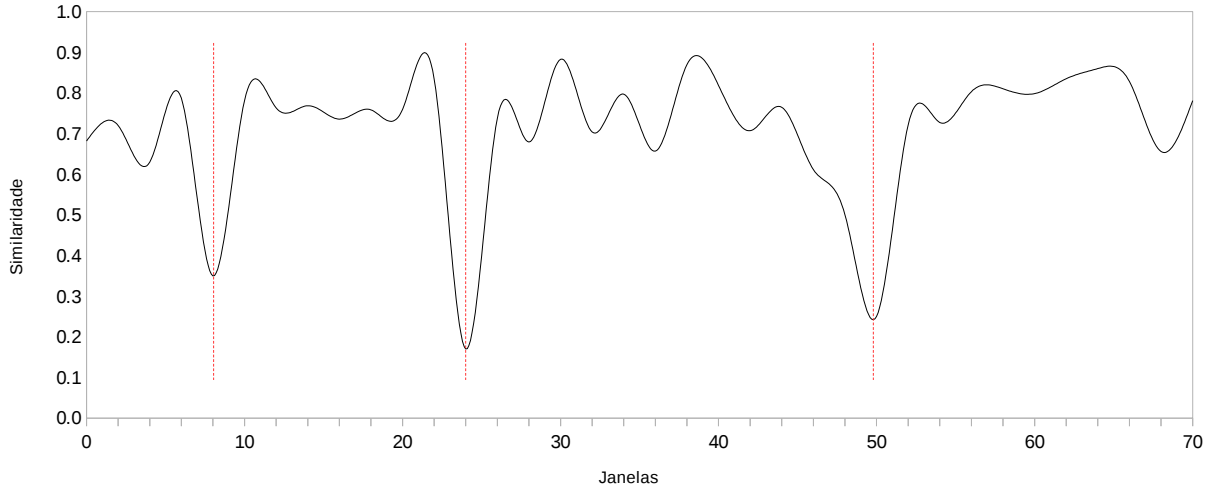

Figura 2 – Curva de dissimilaridades entre blocos de texto adjacentes. As linhas pontilhadas representam diminuições de similaridade que indicam limites entre segmentos.

O TextTiling apresenta como vantagens a facilidade de implementação e baixa complexidade computacional, favorecendo a implementação de trabalhos similares (NAILI; CHAIBI; GHEZALA, 2016; BOKAEI; SAMETI; LIU, 2015; CHAIBI; NAILI; SAMMOUD, 2014; KERN; GRANITZER, 2009; GALLEY et al., 2003), e usado com base line em outros trabalhos (CARDOSO; PARDO; TABOADA, 2017; DIAS; ALVES; LOPES, 2007). Por outro lado, algoritmos mais complexos, como os baseados em matrizes de similaridade, apresentam acurácia relativamente superior como apresentado posteriormente em (CHOI, 2000; KERN; GRANITZER, 2009; MISRA et al., 2009).

Outro algoritmo frequentemente referenciado na literatura é o C99 (CHOI, 2000) o qual é baseado em uma matriz de *ranking* das similaridades. Embora muitos trabalhos utilizem a coesão léxica do texto, para pequenos segmentos pode não ser confiável, pois a ocorrência adicional de uma palavra pode causar certo impacto e alterar o cálculo da similaridade. Além disso, o estilo da escrita normalmente não é constante em todo o texto. Por exemplo, textos iniciais dedicados a introdução costumam apresentar menor coesão do que trechos dedicados a um tópico específico. Portanto, comparar a similaridade entre trechos de diferentes regiões não é apropriado. Devido a isso, as similaridades não podem ser comparadas em valores absolutos. Então, contorna-se esse problema fazendo uso de matrizes de similaridade para encontrar os segmentos de texto. Para isso, o C99 constrói uma matriz que contém as similaridades de todas as unidades de informação (normalmente sentenças ou parágrafos).

Na Figura 3 é mostrado um exemplo de uma matriz de similaridade onde a intensidade do ponto(i, j) representa a similaridade entre as sentenças i e j. Observa-se

que a matriz é simétrica, assim cada ponto na linha diagonal representa a similaridade quanto $i = j$ (ou seja, com a mesma sentença) e revela quadrados com maior concentração de pontos ao longo da diagonal. Essas regiões indicam porções de texto com maior coesão léxica.

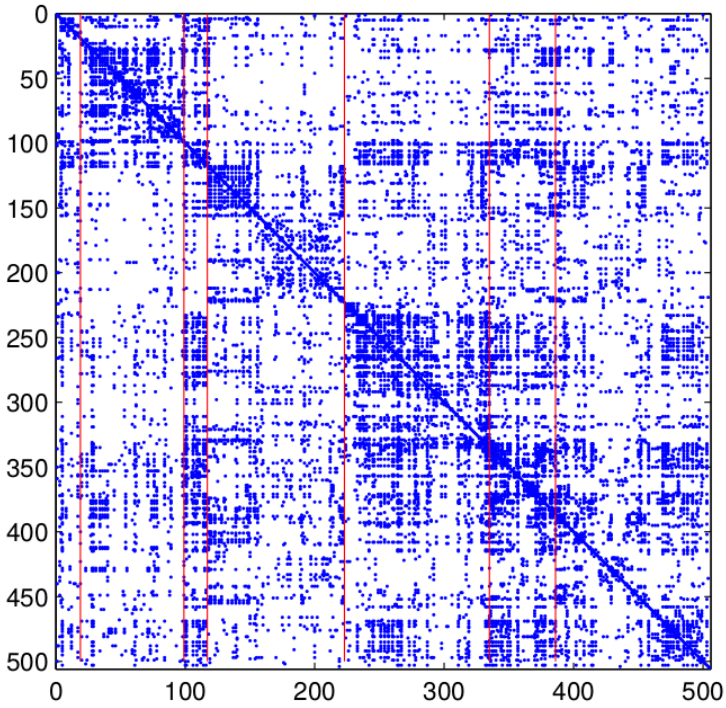

Figura 3 – *DotPlot* da similaridade entre sentenças onde as linha verticais representam segmentos reais (EISENSTEIN; BARZILAY, 2008).

Em seguida, cada valor na matriz de similaridade é substituído por seu *ranking local*. Para cada elemento da matriz, seu *ranking* será o número de elementos vizinhos com valor de similaridade menor que o seu. Assim, cada elemento é comparado com seus vizinhos dentro de uma região denominada máscara. Na Figura 2.1 é destacado um quadro 3 x 3 de uma matriz em que cada elemento é a similaridade entre duas unidades de informação. Tomando como exemplo o elemento com valor $0,5$, a mesma posição na matriz de *rankings* terá o valor 4, pois esse é o número de vizinhos com valores inferiores a $0,5$ dentro do quadro analisado na matriz de similaridades. Da mesma forma, na Figura 2.1 para o valor $0,2$ a matriz de *rankings* conterá o valor 1 na mesma posição. Após a construção da matriz de ranking obtêm-se um maior contraste entre facilitando a detecção de limites quando a queda de similaridade entre sentenças é mais sutil.

Finalmente, com base na matriz de *ranking*, o C99 utiliza um método de *clustering* baseado no algoritmo *DotPloting* (REYNAR, 1998) que usa regiões com maior densidade em uma matriz de similaridades para determinar como os segmentos estão distribuídos. Um segmento é definido por duas sentenças i e j que representam uma região quadrada ao longo da diagonal da matriz. Calcula-se a densidade dessa região como mostrado na

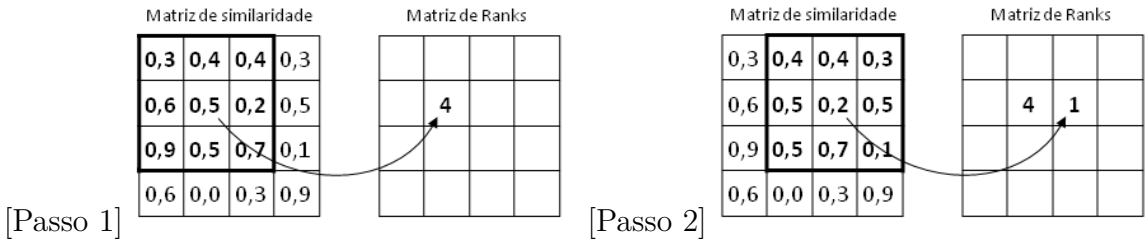

Figura 4 – Exemplo de construção de uma matriz de rankings.

Equação 2.2. Seja $s_{i,j}$ a somatória dos *rakings* de um segmento e $a_{i,j}$ sua área interior. Seja $B = \{b1, ..., b_m\}$ a lista de m segmentos e s_k e a_k são a somatória dos valores dos rankings e a área de um segmento k em B. Então, a densidade é computada por:

$$D = \frac{\sum_{k=1}^{m} s_k}{\sum_{k=1}^{m} a_k} \tag{2.2}$$

O processo incia com um único segmento formado por todas as sentenças do documento e o divide recursivamente em m segmentos. Cada passo divide um dos segmentos em B no ponto (i, j) que maximiza D (Equação 2.2). O processo ser repte até atingir o número de segmentos desejados ou um limiar de similaridade.

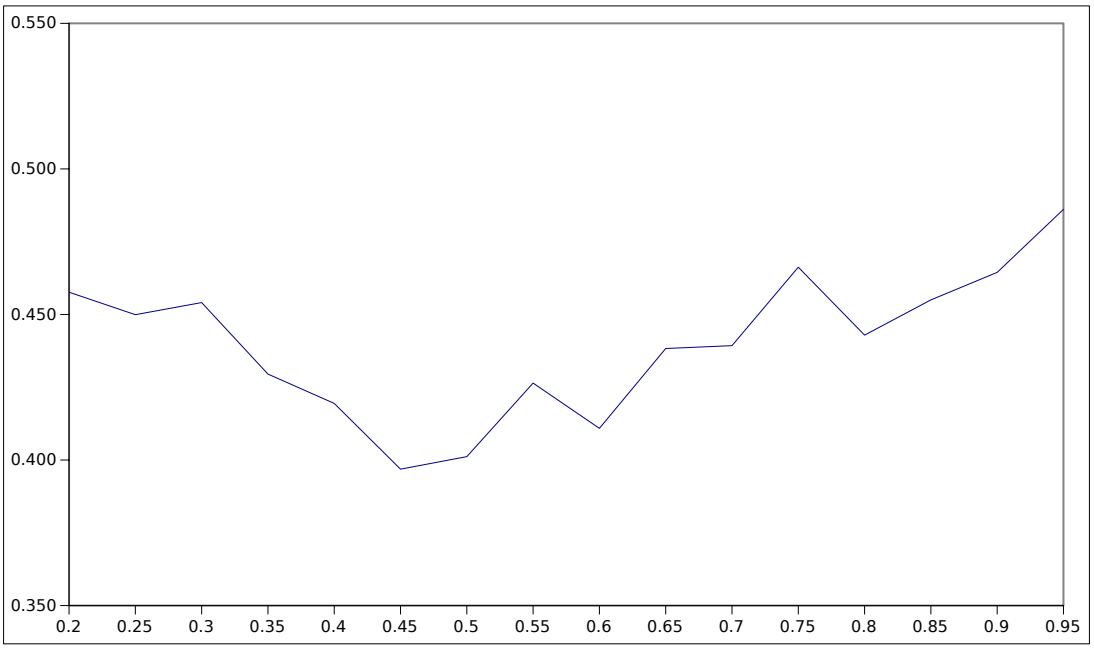

Figura 5 – Influência da quantidade de segmentos em *WindowDiff*

Os métodos baseados em coesão léxica que utilizam métricas como cosseno quantificam a similaridade entre sentenças baseando-se apenas na frequência das palavras, Essa abordagem, ignora certas características do texto que podem dar pistas sobre a estrutura do texto. Por exemplo, frases como "Prosseguindo", "Dando continuidade", "Ao final da reunião"podem dar "pistas"de inicio ou final de segmento. A fim de aproveitar

esses indicadores, usa-se um framework bayesiano que permite incorporar fontes externas ao modelo. O método BayesSeg (EISENSTEIN; BARZILAY, 2008) aborda a coesão léxica em um contexto bayesiano onde as palavas de um segmento surgem de um modelo de linguagem multinomial o qual é associado a um assunto.

Essa abordagem é similar à métodos probabilísticos de extração de tópicos como o Latent Dirichlet Allocation (LDA) (BLEI; NG; JORDAN, 2003), com a diferença que ao invés de atribuir tópicos ocultos a cada palavra, esses são usados para segmentar o documento. Nesse sentido, detecta-se um limite entre sentenças quando a distribuição de tópicos entre elas for diferente.

Baseia-se na ideia que alguns termos são usados em tópicos específicos enquanto outros são neutros em relação aos tópicos do documento e são usados para expressar uma estrutura do documento, ou seja, as "frases-pista"vem de um único modelo generativo. A fim de refletir essa ideia, o modelo é adaptado para influenciar a probabilidade da sentença de ser uma final ou início de segmento conforme a presença de "frases pista".

2.1.1 Medidas de Avaliação

As medidas de avaliação tradicionais como precisão e revocação são permitem medir o desempenho de modelos de Recuperação de Informação e Aprendizado de Máquina por meio da comparação dos valores produzidos pelo modelo com os valores observados em uma referência. Usa-se uma tabela, chamada matriz de confusão, para visualizar o desempenho de um algoritmo. Na Tabela 1 é apresentada uma matriz de confusão para duas classes (Positivo e Negativo).

	Predição Positiva	Predição Negativa
Positivo real	VP (Verdadeiro Positivo)	FN (Falso Negativo)
Negativo real	FP (Falso Positivo)	VN (Verdadeiro Negativo)

Tabela 1 – Matriz de confusão.

No contexto de segmentação textual, um falso positivo é um limite identificado pelo algoritmo que não corresponde a nenhum limite na segmentação de referência, ou seja, o algoritmo indicou que em determinado ponto há uma quebra de segmento, mas na segmentação de referência, no mesmo ponto, não há. De maneira semelhante, um falso negativo é quando o algoritmo não identifica um limite existente na segmentação de referência, ou seja, em determinado ponto há, na segmentação de referência, um limite entre segmentos, contudo, o algoritmo não o identificou. Um verdadeiro positivo é um ponto no texto indicado pelo algoritmo e pela segmentação de referência como uma quebra de segmentos, ou seja, o algoritmo e a referência concordam que em determinado ponto há uma transição de assunto. Na avaliação de segmentadores, não há o conceito de verdadeiro negativo. Este seria um ponto no texto indicado pelo algoritmo e pela segmentação de

referência onde não há uma quebra de segmentos. Uma vez que os algoritmos apenas indicam onde há um limite, essa medida não é necessária.

Nesse sentido, a precisão indica a proporção de limites corretamente identificados pelo algoritmo, ou seja, correspondem a um limite real na segmentação de referência. Porém, não diz nada sobre quantos limites reais existem. É calculada dividindo-se o número de limites identificados automaticamente pelo número de candidatos a limite (Equação 2.3).

$$Precisão = \frac{VP}{VP + FP} \tag{2.3}$$

A revocação, é a proporção de limites verdadeiros que foram identificados pelo algoritmo. Porém não diz nada sobre quantos limites foram identificados incorretamente. É calculada dividindo-se o número de limites identificados automaticamente pelo número limites verdadeiros (Equação 2.4).

$$Revocação = \frac{VP}{VP + FN} \tag{2.4}$$

Existe uma relação inversa entre precisão e revocação. Conforme o algoritmo aponta mais segmentos no texto, este tende a melhorar a revocação e ao mesmo tempo, reduzir a precisão. Esse problema de avaliação pode ser contornado utilizado a medida F^1 que é a média harmônica entre precisão e revocação onde ambas tem o mesmo peso (Equação 2.5).

$$F^1 = \frac{2 \times Precisão \times Revocação}{Precisão + Revocação} \tag{2.5}$$

As medidas de avaliação tradicionais, precisão e revocação, podem não ser confiáveis, por não considerarem a distância entre os limites, mas penalizam o algoritmo sempre que um limite que não coincide perfeitamente com a referência. Essas medidas podem ser mais adequadas quando necessita-se de segmentações com maior exatidão. Em outras palavras, computam apenas os erros do algoritmo quando se detecta falsos positivos ou falsos negativos, o que nesse contexto de segmentação textual pode não ser suficiente, dado a subjetividade da tarefa. Além dessas medidas, que consideram apenas se um segmento foi perfeitamente definido conforme uma referência, pode-se também considerar a distância entre o segmento extraído automaticamente e o segmento de referência (KERN; GRANITZER, 2009). Chama-se *near misses* o caso em que um limite identificado automaticamente não coincide exatamente com a referência, mas é necessário considerar a proximidade entre eles.

Na Figura 6 é apresentado um exemplo com duas segmentações extraídas automaticamente e uma referência. Em ambos os casos não há nenhum verdadeiro positivo, o que implica em zero para os valores de precisão, acurácia, e revocação, embora o resultado do

algoritmo A possa ser considerado superior ao primeiro se levado em conta a proximidade dos limites.

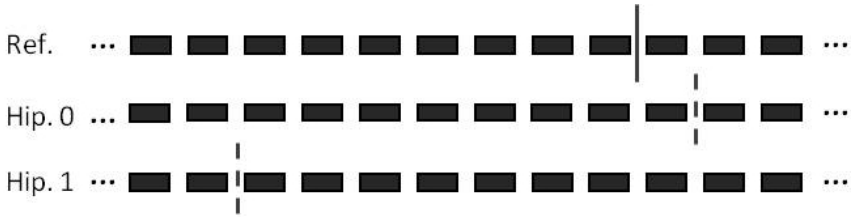

Figura 6 – Exemplos de *near missing* e falso positivo puro. Os blocos indicam uma unidade de informação e as linha verticais representam uma transição de assunto.

Considerando o conceito de *near misses*, algumas soluções foram propostas. Proposta por (BEEFERMAN; BERGER; LAFFERTY, 1999), P_k atribui valores parciais a *near misses*, ou seja, limites sempre receberão um peso proporcional à sua proximidade, desde que dentro de um janela de tamanho k. Para isso, esse método move uma janela de tamanho k ao longo do texto. A cada passo verifica, na referência e na hipótese, se as extremidades (a primeira e última sentença) da janela estão ou não dentro do mesmo segmento, então, penaliza o algoritmo caso este não concorde com a referência. Ou seja, dado duas palavras de distância k, o algoritmo é penalizado caso não concorde com a segmentação de referência se as palavras estão ou não no mesmo segmento. Dadas uma segmentação de referência ref e uma segmentação automática hyp, ambas com N sentenças, P_k é computada como:

$$P_k(ref, hyp) = \frac{1}{N-k} \sum_{i=1}^{N-k} (\delta_{ref}(i, i+k) \bar{\oplus} \delta_{hyp}(i, i+k)) \tag{2.6}$$

onde $\delta_S(i, j)$ é a função indicadora que retora 1 se as sentenças i e j estão no mesmo segmento e 0 caso contrário, $\bar{\oplus}$ é o operador `XNOR` (ou exclusivo) que retorna 1 se ambos os argumentos forem diferentes. O valor de k é calculado como a metade da média dos comprimentos dos segmentos reais. Como resultado, é retornada a dissimilaridade entre as segmentação calculada pela contagem de discrepâncias divida pela quantidade de segmentações analisadas. Essa medida pode ser interpretada como a probabilidade de duas sentenças extraídas aleatoriamente pertencerem ao mesmo segmento.

WindowDiff (PEVZNER; HEARST, 2002) é uma medida alternativa à P_k. De maneira semelhante, move uma janela pelo texto e penaliza o algoritmo sempre que o número de limites proposto pelo algoritmo não coincidir com o número de limites esperados para aquela janela. Ou seja, o algoritmo é penalizado quando não concordar com a segmentação de referência quanto ao número de segmentos na janela. Mais formalmente, para cada intervalo k, compara o número de segmentos obtidos pela referência r_i com o obtido pelo algoritmo a_i e penaliza o algoritmo se $r_i \neq a_i$. Na Equação 2.7 é mostrada a

definição de WindowDiff onde $b(i, i+k)$ representa o número de limites entre as sentenças i e $i+k$ e N, o total de sentenças no texto.

$$WindowDiff(ref, hyp) = \frac{1}{N-k} \sum_{i=1}^{N-k} (|b(ref_i - ref_{i+k}) - b(hyp_i - hyp_{i+k})| > 0) \quad (2.7)$$

Assim, consegue manter a sensibilidade a *near misses* e além disso, considerar o tamanho das janelas. A fim de melhor equilibrar o peso dos falsos positivos em relação a *near misses*, dobra-se a penalidade para falsos positivos, evitando-se a supervalorização dessa medida.

As medidas *WindowDiff* e P_k, consideram a quantidade e proximidade entre os limites, sendo mais tolerantes a pequenas imprecisões. Essa é uma característica desejável, visto que as segmentações de referência possuem diferenças consideráveis. *WindowDiff* equilibra melhor os falsos positivos em relação a *near misses*, ao passo que P_k os penaliza com peso maior. Isso significa que segmentadores melhores avaliados em P_k ajudam a selecionar as configurações que erram menos ao separar trechos de texto com o mesmo assunto, enquanto *WindowDiff* é mais tolerante nesse aspecto. De maneira geral, observa-se melhores resultados de *WindowDiff* quando os algoritmos aproximam a quantidade de segmentos automáticos da quantidade de segmentos da referência. Por outro lado, P_K avalia melhor as configurações que retornam menos segmentos. Contudo, não é possível definir um valor adequado, uma vez que os segmentadores humanos frequentemente apontam segmentações diferentes.

Ao final do processo de segmentação, são produzidos fragmentos de documentos, aqui chamados de subdocumentos. Esses subdocumentos contém um texto, assim como no documento original, em um estágio de processamento inicial, pois ainda não estão estruturados. Ocorre que as técnicas de aprendizado de máquina exigem uma representação estruturada dos textos conforme será visto na Seção 2.2.

2.2 Representação de Textos

Uma das formas mais comuns para que a grande maioria dos algoritmos de aprendizado de máquina possa extrair padrões das coleções de textos é a representação no formato matricial conhecido como Modelo Espaço Vetorial (*Vectorial Space Model* - VSM) (REZENDE, 2003), onde os documentos são representados como vetores em um espaço Euclidiano T-dimensional em que cada termo extraído da coleção é representado por uma dimensão. Assim, cada componente de um vetor expressa a relação entre os documentos e as palavras. Essa estrutura é conhecida como *document-term matrix* ou matriz documento-termo. Uma das formas mais populares para representação de textos é conhecida como *Bag Of Words* a qual é detalhada a seguir.

2.2.1 *Bag Of Words*

Após a etapa de pré-processamento, onde as *stop words* foram removidas e as palavras reduzidas ao seus radicais (*stemming*), tem-se uma versão reduzida, com menos atributos, dos dados originais. Essa versão pode ser facilmente convertida em uma tabela ou matriz documento-termo. Essa representação, conhecida como *bag-of-words*, onde cada termo é transformado em um atributo (*feature*) (REZENDE, 2003). Essa representação é mostrada pela Tabela 2.

	t_1	t_2	t_j	...	t_n
d_1	a_{11}	a_{12}	a_{1j}	...	a_{1n}
d_2	a_{21}	a_{22}	a_{2j}	...	a_{2n}
d_i	a_{i1}	a_{i2}	a_{ij}	...	a_{in}
...	...	...	...	...	...
d_m	a_{m1}	a_{m2}	a_{mj}	...	a_{mn}

Tabela 2 – Coleção de documentos na representação *bag-of-words*

Essa forma de representação sintetiza a base de documentos em um contêiner de palavras, ignorando a ordem em que ocorrem, bem como pontuações e outros detalhes, preservando apenas o peso de determinada palavra nos documentos. É uma simplificação de toda diversidade de informações contidas na base de documentos sem o propósito de ser uma representação fiel do documento, mas oferecer a relação entre as palavras e os documentos a qual é suficiente para a maioria dos métodos de aprendizado de máquina (REZENDE, 2003).

Nessa representação, a_{ij} é o peso do termo j no documento i e indica a sua relevância dentro da base de documentos. As medidas mais tradicionais para o cálculo desses pesos são a binária, onde o termo recebe o valor 1 se ocorre em determinado documento ou 0 caso contrário; *document frequency*, que é o número de documentos no qual um termo ocorre; *term frequency - tf*, atribui-se ao peso a frequência do termo dentro de um determinado documento; *term frequency-inverse document frequency, tf-idf*, pondera a frequência do termo pelo inverso do número de documentos da coleção em que o termo ocorre.

2.3 Modelos de Extração de Tópicos

Os modelos de extração de tópicos são abordagens não-supervisionadas que visam descobrir padrões latentes nas relações entre os documentos e seus termos. Baseiam-se na premissa de que um documento é produzido a partir de tópicos previamente definidos que determinam os termos a serem utilizados em um documento. Nesse contexto, um documento é uma mistura de tópicos onde cada termo presente no documento pode ser associado a um tópico. Um tópico por sua vez, é uma estrutura com valor semântico que representada por um conjunto de termos e seus pesos que indicam o quão significante esses

termos são para um assunto e pode ser útil para o entendimento do tema ao qual o tópico trata (STEYVERS; GRIFFITHS, 2007; BLEI, 2012).

Para descobrir esses tópicos, algumas técnicas foram propostas. Em termos de metodologia, a maioria dos trabalho enquadram-se em duas principais categorias, os modelos não-probabilísticos e os modelos probabilísticos.

Os modelos não-probabilísticas baseiam-se em técnicas de fatoração de matrizes, onde a matrix documento-termo é projetada em um espaço com menor dimensionalidade chamado *Latent Semantic Space*. Seja $d \in D = \{d_1, \cdots, d_n\}$ o vetor que representa a coleção de documentos, $t \in T = \{t_1, \cdots, t_m\}$ seus termos distintos e $z \in Z = \{z_1, \cdots, z_k\}$ seus tópicos. Esses métodos aprendem decompondo a matriz documento-termo W, em duas matrizes Z e A, tal que a resultante de ZA seja uma aproximação da matriz W original. Mais formalmente tem-se:

$$Z \cdot A = \hat{W} \approx W \tag{2.8}$$

A matriz A corresponde a matriz documento-tópico e possui dimensão $k \times n$. Z corresponde a matriz termo-tópico e possui dimensão $m \times k$ onde n é o número de termos, m é o número de documentos da coleção e k é a quantidade de tópicos a serem extraídos. Uma vez que $k \ll n, m$, então A e Z são menores que a matriz de entrada, o que resulta em uma versão comprimida da matriz original, pois $k \cdot n + m \cdot k \ll n \cdot m$. Ao final, obtém-se uma representação documento-tópico que atribui um peso para cada tópico em cada documento da coleção e uma representação termo-tópico que representa a probabilidade de ocorrência de um termo em um documento dado que o tópico está presente no documento.

Nesse sentido, o *Latente Semantic Indexing* (LSI) (DEERWESTER et al., 1990) usa a técnica chamada *Singular Value Decomposition* (SVD) para encontrar padrões no relacionamento entre assuntos e termos em uma coleção de texto não estruturada. Entretanto, esse método não fornece uma interpretação para elementos com valores negativos (DEERWESTER et al., 1990) (CHENG et al., 2013).

Outro modelo popular é o *Non-Negative Matrix Factorization* (NMF) (LEE; SEUNG, 1999). Diferente do LSI, no processo de fatoração apenas operações aditivas são permitidas, o que garante que as matrizes resultantes não possuem elementos negativos, permitindo uma interpretação mais intuitiva de seus valores. Além disso, o processo de fatoração proporciona a propriedade de *clustering*, ou seja, agrupar as colunas da matriz W, e dessa forma, oferece a característica interessante de agrupar os documentos da coleção.

Os modelos probabilísticos consideram os documentos como uma mistura de tópicos e um tópico como uma distribuição probabilística sobre os termos. O processo de elaboração do documento a partir desses tópicos é chamado de processo generativo ou modelo generativo, o qual é desconhecido porém pode ser estimado com base nos termos presentes

no documento, também chamados de variáveis observáveis. Assim, o processo de extração de tópicos consiste em estimar o modelo generativo que deu origem ao documento. O PLSA (HOFMANN, 1999) foi um dos primeiros a estender o modelo LSA e formalizar a extração de tópicos probabilísticos. De maneira similar ao LSA, o esse modelo decompõe uma matriz esparsa a fim de reduzir a dimensionalidade. O PLSA cria um modelo estatístico chamado *aspect model* que associa os tópicos as variáveis observáveis atribuindo probabilidades às ligações entre os tópicos e os documentos e entre as palavras e os tópicos. Assim, cada documento pode ser representado como a probabilidade de um tópico estar presente, $P(z|d)$. E a probabilidade de um termo ocorrer dado que um tópico esta presente, $P(t|z)$. Em comparação ao LSA, é considerado uma método mais robusto por proporcionar uma interpretação probabilística. Por outro lado, esse modelo apresenta desvantagens como o número de parâmetros do modelo que cresce linearmente com o número de documentos da coleção que pode ocasionar *overfitting*.

A fim de contornar esses problemas, o LDA (BLEI; NG; JORDAN, 2003) estende o modelo PLSA incorporando um modelo generativo onde os cada tópico obedece à distribuição multivariada de *Dirichlet* o que o torna menos propenso ao *overfitting* e capaz de inferir tópicos a documentos ainda não observados. É referenciado na literatura como estado da arte sobre modelos probabilísticos de extração de tópicos e influencia uma grande quantidade de trabalhos, tornando-se base para novos modelos. No modelo LDA, o processo de geração de palavras se dá em duas etapas:

1. Atribui-se uma distribuição aleatória sobre os tópicos.
2. Para cada termo no documento:
    a) Atribui-se aleatoriamente a um tópico da distribuição obtida na etapa 1;
    b) Seleciona-se aleatoriamente uma palavra do tópico correspondente.

Assim cada documento é associado a múltiplos tópicos com proporções distintas (etapa 1). Cada palavra do documento é obtida de um tópico específico (etapa 2.b) que foi anteriormente obtido a partir da distribuição de tópicos do documento (etapa 2.a). Isso permite ao modelo LDA atribuir, para cada documento, múltiplos tópicos com proporções distintas (BLEI, 2012).

Os modelos de extração de tópicos foram inicialmente propostos para utilização em mineração texto onde são empregados na redução de dimensionalidade, extração de informações em textos, bem como na organização e recuperação de documentos, sendo utilizados para mensurar a relevância de um termo ou conjunto de termos para determinado assunto ou documento. Visto a popularidade nessas tarefas e flexibilidade dos modelos, logo notou-se sua utilidade em outros tipos de dados com atributos discretos como imagens, grafos e genética.

2.4 Trabalhos Relacionados

3 Sistema Proposto

Essa seção apresenta as etapas de desenvolvimento do sistema de recuperação de atas proposto, bem como o seu funcionamento geral, desde a preparação dos documentos até a entrega dos históricos de ocorrência ao usuário. Inicialmente serão descritos a seleção e pré-processamento das atas. Em seguida, será relatado como as técnicas de mineração de texto e recuperação de informação são utilizadas nesse trabalho.

A Figura 7 mostra a visão geral do sistema proposto cujo objetivo é permitir ao usuário consultar uma coleção de documentos de reuniões a fim de obter todo o histórico de ocorrências de um determinado tema relacionado à pesquisa do usuário, podendo identificar nos documentos onde esse tema foi mencionado, bem como se houve uma decisão sobre o tema. Para isso, o sistema é divido em dois módulos principais: módulo de preparação e manutenção e módulo de consulta, os quais serão detalhados nas próximas seções.

A Figura 7 mostra a visão geral do sistema com suas principais entradas e saídas.

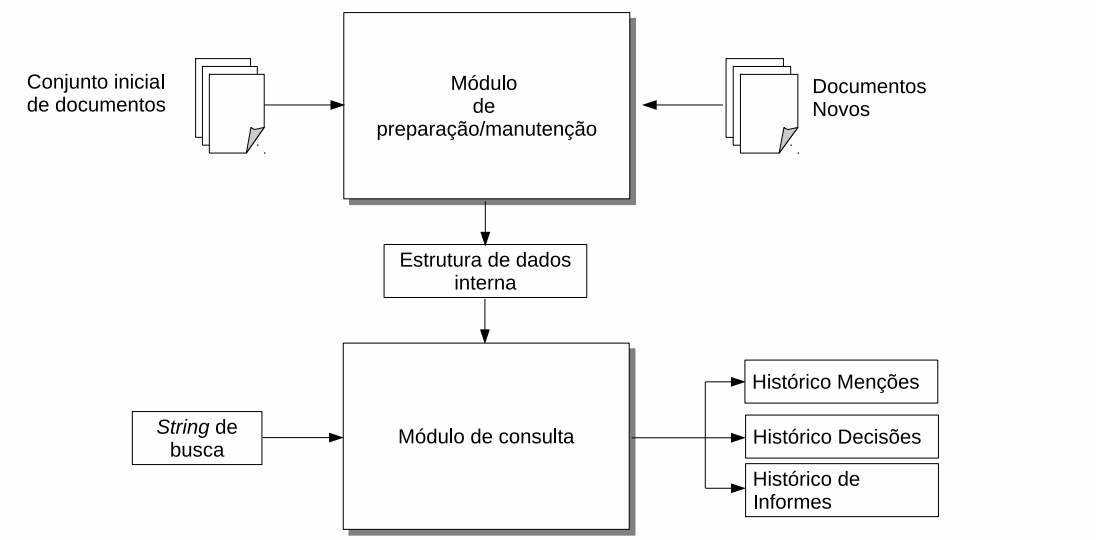

Figura 7 – Visão geral do sistema

3.1 Módulo de preparação e manutenção

O módulo de preparação e manutenção tem como funções principais dividir cada ata em em segmentos de texto que contêm um assunto predominante, e separá-los em categorias por meio de técnicas de extração tópicos e classificação. Além disso, produz uma estrutura de dados que registra quais assuntos foram tratados na reunião, bem como o trecho do documento onde é discutido. A seguir são apresentadas as etapas do módulo de

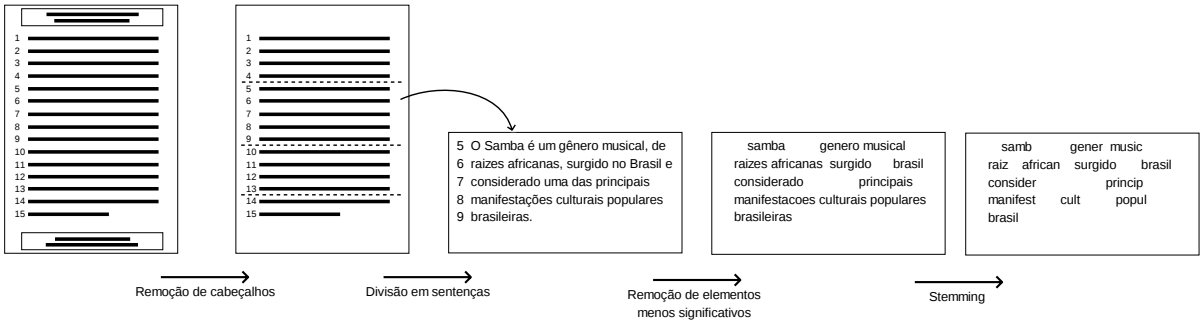

Figura 8 – Etapa de pré-processamento

preparação e manutenção desde a preparação dos documentos até a entrega da estrutura interna ao módulo de consulta.

3.1.1 Preparação dos documentos

As atas são normalmente armazenadas em arquivos binários do tipo *pdf*, *doc*, *docx* ou *odt*. As atas devem ser pré-processadas e estruturadas para que posam ser aplicados métodos de MI e RI. Inicialmente, o texto puro é extraído e passa por processos de transformação conforme apresentados a seguir.

A Figura 8 mostra a etapa de preparação de um documento em português que inclui a remoção de elementos menos significativos e a identificação de sentenças e segmentos.

1. Remoção de cabeçalhos e rodapés: as atas contém trechos que podem ser considerados pouco informativos e descartados durante o pré-processamento, como cabeçalhos e rodapés que se misturam aos tópicos tratados na reunião, podendo ser inseridos no meio de um tópico prejudicando tanto os algoritmos de MT e RI, quanto a leitura do texto pelo usuário. Um cabeçalho é a porção de texto que inicia cada página do documento e, de forma semelhante, um rodapé e a porção que as encerra. Detecta-se os cabeçalhos e os rodapés sempre que há uma repetição das primeiras e últimas palavras do documento.

2. Identificação de finais sentenças: Ao considerar intuitivamente que uma sentença seja uma sequência de palavras entre sinais de pontuação como “.”, “!” e “?”, alguns erros poderiam ocorrer quando esses tiverem outra função dentro do texto como em abreviações, endereços de internet e datas. Outro problema seriam frases curtas com poucas palavras e que não expressam um conceito completo, mas parte dele. Devido ao estilo de pontuação desses documentos, como encerrar sentenças usando um *“;”* e inserção de linhas extras, foram usadas as regras especiais para identificação de finais

de sentença. No Algoritmo 1 é mostrado como cada *token* é identificado e marcado com final de sentença.

Algoritmo 1: Identificação de finais de sentença

Entrada : Texto
Saída : Texto com identificações de finais de sentença

para todo *token, marcá-lo como final de sentença se:*
  Terminar com um `!`
  Terminar com um `.` e não for uma abreviação
  Terminar em `.?;` e:
    For seguido de uma quebra de parágrafo ou tabulação
    O próximo *token* iniciar com `({["'`
    O próximo *token* iniciar com letra maiúscula
    O penúltimo caracter for `)}]"'`
fim

3. Redução de termos: Removeu-se do as palavras que não contribuem para a distinção do texto em tópicos ou categorias, as quais são chamadas de *stop words*. Palavras como artigos, preposições, pronomes, verbos de estado[1]. Trata-se também como *stop words* as palavras de uso muito frequente dentro de um determinado domínio as quais não são capazes de discriminar documentos, portanto também não devem fazer parte dos atributos (REZENDE, 2003). Para removê-las, as letras foram convertidas em caixa baixa e usou-se uma lista de 438 palavras para identificá-las. Além disso, eliminou-se a acentuação, sinais de pontuação, numerais e todos os *tokens* menores que três caracteres.

4. *Stemming*: extraiu-se o radical de cada palavra. Para isso, aplicou-se o algoritmo *Orengo* para remoção de sufixos (ALVARES; GARCIA; FERRAZ, 2005).

3.1.1.1 Segmentação

Como já mencionado, uma ata registra a sucessão de assuntos discutidos em uma reunião, porém apresenta-se com poucas quebras de parágrafo e sem marcações de estrutura, como capítulos, seções ou quaisquer indicações sobre o assunto do texto. Portanto, faz-se necessário descobrir quando há uma mudança de assunto no texto da ata. Para essa tarefa, as técnicas de segmentação de texto recebem uma lista de sentenças, da qual considera cada ponto entre duas sentenças como candidato a limite, ou seja, um ponto onde há transição entre assuntos (BOKAEI; SAMETI; LIU, 2015; BOKAEI; SAMETI; LIU, 2016; MISRA et al., 2009; SAKAHARA; OKADA; NITTA, 2014).

Entre os principais trabalhos da literatura podemos citar o *TextTiling* (HEARST, 1994) e o *C99* (CHOI, 2000) são considerados um dos primeiros mais influentes sendo

[1] Apresentam uma situação inativa, onde o verbo não expressa uma alteração, mas apenas uma propriedade ou condição dos envolvidos.

utilizados com base lines em trabalhos recentes(CHAIBI; NAILI; SAMMOUD, 2014; NAILI; CHAIBI; GHEZALA, 2016; CARDOSO; PARDO; TABOADA, 2017)

3.1.1.2 Segmentação de Referência

Para que se possa avaliar um segmentador automático de textos é preciso uma referência, isto é, um texto com os limites entre os segmentos conhecidos. Essa referência, deve ser confiável, sendo uma segmentação legítima que é capaz de dividir o texto em porções relativamente independentes, ou seja, uma segmentação ideal. A fim de obter um conjunto de documentos segmentados que possam servir como referência na avaliação, os documentos coletados foram segmentados manualmente por dois coordenadores de curso que participam de reuniões. Para isso, utilizou-se um *software*, desenvolvido com esse objetivo especifico, que permitiu aos voluntários visualizar um documento, e indicar livremente as divisões entre segmentos. Com o uso desse *software* foram coletados os dados de seis atas segmentadas pelos participantes das reuniões, os quais serviram como referência para a avaliação dos algoritmos. O *software* desenvolvido para segmentação manual está disponível para utilização e consulta em

Os arquivos gerados foram tratados para que os segmentos sempre terminem em uma sentença reconhecida pelo algoritmo, uma vez que as sentenças são a unidade mínima de informação nesse trabalho.

A Tabela 3 contém, para cada ata, a quantidade de sentenças e a quantidade de segmentos identificadas pelos participantes.

Ata	**Sentenças**	**Participante 1**	**Participante 2**
Ata 1	18	7	15
Ata 2	26	9	20
Ata 3	24	7	15
Ata 4	32	9	17
Ata 5	25	11	17
Ata 6	10	4	9

Tabela 3 – Quantidade de sentenças e segmentos de referência por ata.

3.1.2 Configuração experimental

O *TextTiling* permite ajustarmos dois parâmetros, sendo o tamanho da janela e o passo. Por meio de testes empíricos escolheu-se os valores os valores 20, 40 e 60 para o tamanho da janela e 3, 6, 9 e 12 para o passo. Gerando ao final 20 configurações.

O *C99* permite o ajuste de três parâmetros, sendo, o primeiro a quantidade segmentos desejados, uma vez que, não se conhece o número ideal de segmentos e os documentos não apresentam muitos candidatos, calculou-se uma proporção dos candidatos

a limite. Para isso atribuiu-se os valores 0,2; 0,4; 0,6; 0,8. O segundo parâmetro, o tamanho do quadro utilizado para gerar a matriz de ranking, atribuiu-se os valores 9 e 11, sendo 11 o valor padrão da apresentado pelo autor. O algoritmo permite ainda indicar se as sentenças serão representados por vetores contendo a frequência ou o peso de cada termo. Ambas as representações foram utilizadas. Considerando todos os parâmetros, foram geradas 16 configurações para o algoritmo *C99*.

3.1.2.1 Critérios de avaliação

Para fins de avaliação desse trabalho, um bom método de segmentação é aquele cujo resultado melhor se aproxima de uma segmentação manual, sem a obrigatoriedade de estar perfeitamente alinhado com tal. Ou seja, visto o contexto das atas de reunião, e a subjetividade da tarefa, não é necessário que os limites entre os segmentos (real e hipótese) sejam idênticos, mas que se assemelhem em localização e quantidade.

Os algoritmos foram comparados com a segmentação fornecida pelos participantes das reuniões. Calculou-se as medidas mais aplicadas à segmentação textual, P_k e *WindowDiff*. Além dessas, computou-se também as medidas tradicionais acurácia, precisão, revocação e F^1 para comparação com outros trabalhos que as utilizam.

Inicialmente, calculou-se as medidas configurando cada algoritmo conforme mostrado na Subseção 3.1.2, sem aplicar o pré-processamento. O teste de Friedman com pós-teste de Nemenyi foi utilizado para gerar um ranking das melhores configurações para cada medida calculada. Com isso, foi possível descobrir quais valores otimizam um algoritmo para uma medida, desconsiderando o pré-processamento.

A fim de conhecer o impacto do pré-processamento, repetiu-se os testes com o texto pré-processado. Com isso, descobriu-se quais valores otimizam os algoritmos para cada medida, considerando essa etapa.

Com os testes anteriores obteve-se, para cada medida, 4 configurações, levando em conta ambos os algoritmos e a presença ou ausência do pré-processamento. Novamente utilizou-se o teste de Friedman e Nemenyi e descobriu-se, para cada medida, qual configuração a otimiza. Os resultados completos estão disponíveis para consulta em .

3.1.2.2 Resultados

Obteve-se, por meio dos testes estatísticos apresentados, as melhores configurações para as principais medidas de avaliação de segmentadores. Com essas configurações calculou-se a média de cada medida considerando o conjunto de documentos. Na Tabela 4 são apresentadas, as médias obtidas com o *TextTiling* bem como as configurações utilizadas, onde **J** é o tamanho da janela e **P** é o passo.

Uma vez que a coesão léxica é pressuposto de muitas abordagens em segmentação

Medida	Sem Pré-processamento J	P	Média	Com Pré-processamento J	P	Média
P_k	50	9	0,142	50	9	0,144
WindowDiff	50	6	0,387	40	9	0,396
Acurácia	50	6	0,612	40	9	0,603
Precisão	40	9	0,611	50	12	0,613
Revocação	20	3	0,886	20	3	0,917
F^1	30	6	0,605	40	3	0,648

Tabela 4 – Resultados obtidos com o *TextTiling*

textual, fez-se uma análise desses documentos quanto a similaridade dos termos ao longo do texto. Verificou-se que a técnica de janelas deslizantes empregada pelo TextTiling encontra os vales que indicam transições entre segmentos, contudo ao comparar esses vales com a segmentação de referência, nota-se que a maioria dos limites coincide ou estão próximos aos vales, porém há casos onde a referência indica limites em trechos com alta coesão léxica e outros onde a queda da coesão, indicada por vales, não coincide com nenhum limite de referência.

Na Figura 9 é apresentado a variação da coesão léxica ao longo de uma ata e a segmentação obtida pelo *TextTiling* usando tamanho de janela igual a 50 e passo 9. A linha horizontal representa a variação da coesão léxica e as linha verticais azuis e vermelhas representam os limites entre segmentos atribuidos pela referência e pelo algoritmo respectivamente.

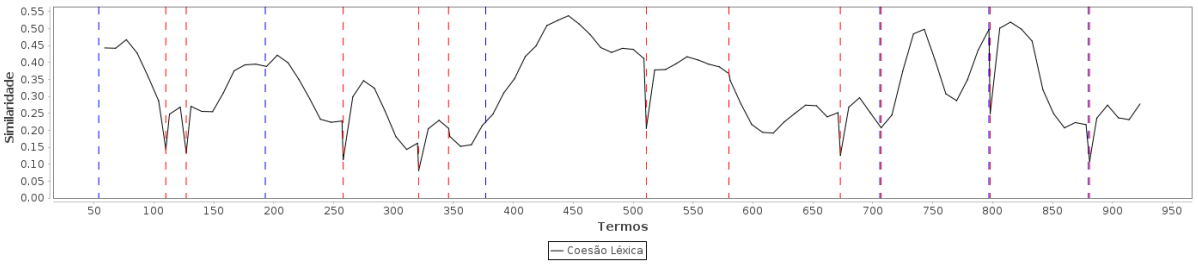

Figura 9 – Variação da coesão léxica ao longo de uma ata junto a uma segmentação automática em contraste com uma segmentação de referência.

Analisou-se também o desempenho da mesma técnica aplicada a um texto contínuo extraído de artigo da Interent que descreve seis gêneros musicais brasileiros um após um outro separados em seções. Ao observar a Figura 10, nota-se que os vales são mais definidos e a maioria dos segmentos coincidem ou estão próximos a segmentação de referência. A segmentação de refência possui sete segementos que separam uma introdução do assuntos e respeitam cada uma das subseções que tratam de um gênero músical. Obtem-se nesse cenário uma eficiência maior em relação a segmentação da ata, o que sugere que textos organizados em seções podem ter melhores benefícios com técnicas baseadas em coesão léxica que as atas, onde esse fator é menos significativo.

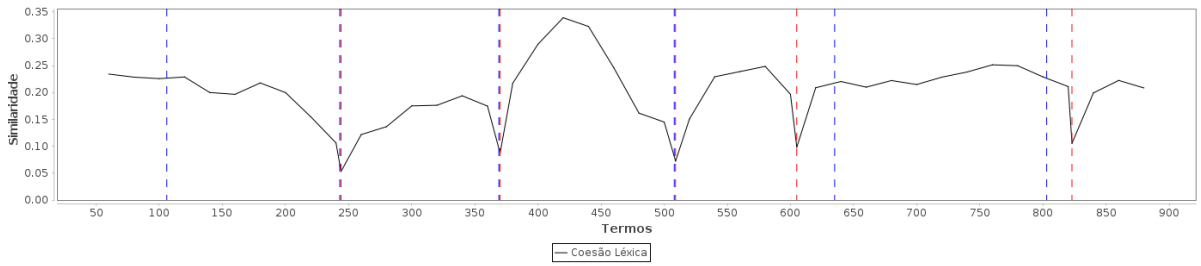

Figura 10 – Variação da coesão léxica ao longo de um artigo melhor estruturado em seções junto a uma segmentação automática em contraste com uma segmentação de referência.

Na Tabela 5 são apresentadas, as médias obtidas com o *C99* bem como as configurações utilizadas, onde **S** é a proporção de segmentos em relação a quantidade de candidatos, **M** é o tamanho do quadro utilizado para criar a matriz de *rankings* e **W** indica se os segmentos são representados por vetores contendo a frequência ou um peso das palavras.

	Sem Pré-processamento				Com Pré-processamento			
Medida	**S**	**M**	**W**	**Média**	**S**	**M**	**W**	**Média**
P_k	20	9	Sim	0,134	20	11	False	0,116
WindowDiff	60	9	Sim	0,411	60	9	Sim	0,390
Acurácia	60	9	Sim	0,588	60	9	Sim	0,609
Precisão	40	9	Sim	0,645	20	11	False	0,720
Revocação	80	9	Sim	0,869	80	11	Sim	0,897
F^1	80	9	Sim	0,638	80	11	Sim	0,655

Tabela 5 – Resultados obtidos com o *C99*

Verificou-se que o *C99* obteve melhor desempenho em acurácia, precisão, F^1, P_k e *WindowDiff*, em relação ao *TextTiling*, enquanto este obteve o melhor desempenho em revocação. De maneira geral, o algoritmo *C99* apresenta melhores resultados em relação ao *TextTiling*, contudo testes estatísticos realizados indicaram que não houve diferença significativa entre os métodos.

A avaliação final foi feita pela comparação dos algoritmos usando as medidas *Pk* e *WindowDiff*. É apresentada também, para fins de comparação, as medidas tradicionais acurácia, precisão, revocação e F^1, entretanto, nesse contexto, essas medidas são menos significativa que Pk e WindowDiff, conforme já mencionado na Seção **??**. A Tabela 6 contém as médias com cada algoritmo. Vale lembrar que P_k e *WindowDiff* são medidas de dissimilaridade, ou seja, os valores menores significam melhores resultados.

Na Figura 11 é apresentada a performance dos algoritmos nas medidas tradicionais. Observa-se valores altos de revocação para a segmentação por sentenças, pois é atribuído um limite a todo candidato a final de segmento, o que resulta no valor máximo para revocação. De maneira semelhante, o comportamento do *TextTiling* gera mais segmentos

Método	**Pk**	**WD**	**A**	**P**	**R**	**F1**	**Segmentos**
Sentenças	0.320	0.502	0.498	0.498	**1.000**	**0.642**	22.083
TextTiling	0.275	0.469	0.531	0.514	0.937	0.640	19.583
C99	0.142	0.426	0.574	0.601	0.473	0.506	8.167
BayesSeg	0.148	0.414	0.586	0.599	0.526	0.528	8.750
MinCut	0.226	0.532	0.468	0.464	0.438	0.432	10.333
TextSeg	**0.085**	**0.387**	**0.613**	**0.714**	0.412	0.497	5.167

Tabela 6 – Melhores resultados obtidos.

em relação aos demais, e com isso tem-se valores maiores de revocação, o que pode ser contornado configurando o algoritmo com passos maiores, ou ainda, sobre-escrevendo a função que calcula os *depth scores* para reconhecer vales mais largos.

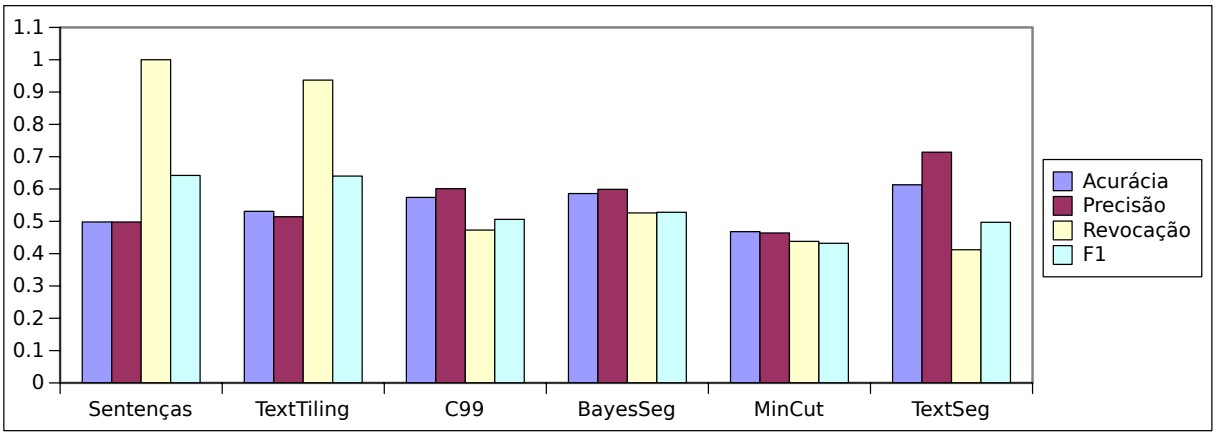

Figura 11 – ...

Na Figura 12 é apresentada a performance dos algoritmos nas medidas P_k e *WindowDiff*. Verifica-se que *TextSeg* apresenta valores de *WindowDiff* próximas ao *C99* e *BayesSeg* e resultados mais significantes quando medidos por P_k em relação aos demais algoritmos.

Após a identificação dos segmentos, o algoritmo retorna uma lista onde cada elemento é um texto com um assunto predominante e será a partir de disso considerado um documento.

3.1.3 Representação Computacional

As etapas anteriores produzem fragmentos de documentos onde o texto esta em um estágio de processamento inicial, com menos atributos que as versões originais, onde cada fragmento está associado a um tema, porém, ainda não estruturado. Ocorre que as técnicas de mineração de texto exigem uma representação estruturada dos textos.

Uma das formas mais comuns é a representação no formato matricial conhecida como Modelo Espaço Vetorial (*Vectorial Space Model* - VSM) (REZENDE, 2003), onde os

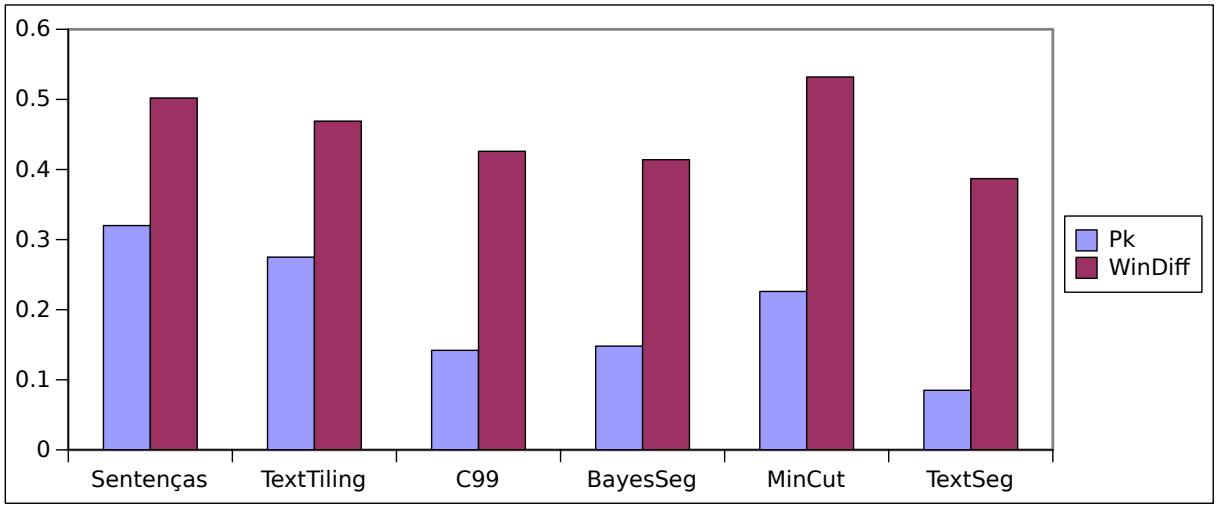

Figura 12 – ...

documentos são representados como vetores em um espaço Euclidiano t-dimensional em que cada termo extraído da coleção é representado por um dimensão. Assim, cada componente de um vetor expressa a relação entre os documentos e as palavras. Essa estrutura é conhecida como *document-term matrix* ou matriz documento-termo. Nesse trabalho a representação empregada é a *Bag Of Words* a qual sintetiza a base de documentos em um contêiner de palavras, ignorando a ordem em que ocorrem, bem como pontuações e outros detalhes, preservando apenas o peso de determinada palavra nos documentos.

3.1.4 Extração de Tópicos

3.2 Módulo Consulta

Uma vez que a estrutura de dados interna contem os assuntos abordados na coleção de documentos, o tipo de ocorrência para cada assunto e o trecho onde se encontram, caberá ao módulo de consulta receber a *string* de consulta do usuário, resgatar os dados desejados e apresentá-los em ordem cronológica, dando condições para o usuário acessar os segmentos encontrados bem como os documentos originais.

3.2.1 Visualização

O usuário final precisa de uma interface adequada para visualizar os resultados da busca considerando-se a relevância dos tópicos selecionados e a sequência cronológica.

Uma boa apresentação deve permitir ao usuário identificar a relevância os resultados e ser relativante independente para compreensão do conteúdo, evitando a leitura do texto completo. Ou seja, o texto de cada tópico apresentado deve ser suficiente para compreensão do assunto mencionado, sem necessidade de visualizar o documento original.

As informações apresentadas, incluem dados obtidos do documento como o nome do arquivo e data do original e o texto onde o assunto é mencionado. Além disso, apresenta-se as informação extraídas pelas técnicas de mineração de texto como os descritores e rótulos.

Para cada busca, é retornada uma lista de resultados ordenados pela relevância com a *string* de entrada, sendo cada item referente a uma menção a um assunto. Um tópico é abordado em diferentes momentos e registrado em atas distintas, onde cada menção é um resultado a ser apresentado.

Como parte da proposta, o sistema apresenta cada resultado dentro de um histórico de menções. Para isso, abaixo do texto é exibida uma linha com links para os resultados que compartilham o mesmo tópico ordenados por data. Os links, ao ser acionado, direciona para o resultado que aponta, além disso, quando o cursor do mouse está sobre o link, é apresentado um pre-visualização do texto. Dessa forma o usuário tem acesso uma interface que lhe fornece uma visão temporal das menções.

3.3 Estudo de caso

3.4 Avaliação

Referências

ALVARES, R. V.; GARCIA, A. C. B.; FERRAZ, I. Stembr: A stemming algorithm for the brazilian portuguese language. In: *Proceedings of the 12th Portuguese Conference on Progress in Artificial Intelligence*. Berlin, Heidelberg: Springer-Verlag, 2005. (EPIA'05), p. 693–701. ISBN 3-540-30737-0, 978-3-540-30737-2. Disponível em: <http://dx.doi.org/10.1007/11595014_67>. Citado na página 23.

BEEFERMAN, D.; BERGER, A.; LAFFERTY, J. Statistical models for text segmentation. *Machine Learning*, v. 34, n. 1, p. 177–210, 1999. ISSN 1573-0565. Disponível em: <http://dx.doi.org/10.1023/A:1007506220214>. Citado na página 14.

BLEI, D. M. Probabilistic topic models. *Commun. ACM*, ACM, New York, NY, USA, v. 55, n. 4, p. 77–84, abr. 2012. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/2133806.2133826>. Citado 3 vezes nas páginas 7, 17 e 18.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. *J. Mach. Learn. Res.*, JMLR.org, v. 3, p. 993–1022, mar. 2003. ISSN 1532-4435. Disponível em: <http://dl.acm.org/citation.cfm?id=944919.944937>. Citado 2 vezes nas páginas 12 e 18.

BOKAEI, M. H.; SAMETI, H.; LIU, Y. Linear discourse segmentation of multi-party meetings based on local and global information. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, IEEE Press, Piscataway, NJ, USA, v. 23, n. 11, p. 1879–1891, nov. 2015. ISSN 2329-9290. Disponível em: <http://dx.doi.org/10.1109/TASLP.2015.2456430>. Citado 2 vezes nas páginas 9 e 23.

BOKAEI, M. H.; SAMETI, H.; LIU, Y. Extractive summarization of multiparty meetings through discourse segmentation. *Natural Language Engineering*, Cambridge University Press, v. 22, n. 1, p. 41–72, 2016. Citado na página 23.

CARDOSO, P.; PARDO, T.; TABOADA, M. Subtopic annotation and automatic segmentation for news texts in brazilian portuguese. *Corpora*, Edinburgh University Press, v. 12, n. 1, p. 23–54, 2017. Citado 2 vezes nas páginas 9 e 24.

CHAIBI, A. H.; NAILI, M.; SAMMOUD, S. Topic segmentation for textual document written in arabic language. *Procedia Computer Science*, v. 35, p. 437 – 446, 2014. ISSN 1877-0509. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877050914010898>. Citado 2 vezes nas páginas 9 e 24.

CHENG, X. et al. Learning topics in short texts by non-negative matrix factorization on term correlation matrix. In: *SDM*. SIAM, 2013. p. 749–757. ISBN 978-1-61197-283-2. Disponível em: <http://dblp.uni-trier.de/db/conf/sdm/sdm2013.html#ChengGLWY13>. Citado na página 17.

CHOI, F. Y. Y. Advances in domain independent linear text segmentation. In: *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2000. (NAACL 2000), p. 26–33. Disponível em: <http://dl.acm.org/citation.cfm?id=974305.974309>. Citado 3 vezes nas páginas 5, 9 e 23.

DEERWESTER, S. et al. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, v. 41, n. 6, p. 391–407, 1990. Citado 2 vezes nas páginas 7 e 17.

DIAS, G.; ALVES, E.; LOPES, J. G. P. Topic segmentation algorithms for text summarization and passage retrieval: An exhaustive evaluation. In: *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 2*. AAAI Press, 2007. (AAAI'07), p. 1334–1339. ISBN 978-1-57735-323-2. Disponível em: <http://dl.acm.org/citation.cfm?id=1619797.1619859>. Citado na página 9.

EISENSTEIN, J.; BARZILAY, R. Bayesian unsupervised topic segmentation. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008. (EMNLP '08), p. 334–343. Disponível em: <http://dl.acm.org/citation.cfm?id=1613715.1613760>. Citado 2 vezes nas páginas 10 e 12.

GALLEY, M. et al. Discourse segmentation of multi-party conversation. In: *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003. (ACL '03), p. 562–569. Disponível em: <http://dx.doi.org/10.3115/1075096.1075167>. Citado na página 9.

HEARST, M. A. Multi-paragraph segmentation of expository text. In: *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 1994. (ACL '94), p. 9–16. Disponível em: <http://dx.doi.org/10.3115/981732.981734>. Citado na página 23.

HOFMANN, T. Probabilistic latent semantic indexing. In: *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA: ACM, 1999. (SIGIR '99), p. 50–57. ISBN 1-58113-096-1. Disponível em: <http://doi.acm.org/10.1145/312624.312649>. Citado 2 vezes nas páginas 7 e 18.

KERN, R.; GRANITZER, M. Efficient linear text segmentation based on information retrieval techniques. *Proceedings of the International Conference on Management of Emergent Digital EcoSystems, MEDES '09*, p. 167–171, 2009. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-74549147972&doi=10.1145%2f1643823.1643854&partnerID=40&md5=1c6f73bc0e07446fcc178440e48bbc40>. Citado 2 vezes nas páginas 9 e 13.

KOZIMA, H. Text segmentation based on similarity between words. In: *Proceedings of the 31st Annual Meeting on Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 1993. (ACL '93), p. 286–288. Disponível em: <http://dx.doi.org/10.3115/981574.981616>. Citado na página 7.

LEE, D.; SEUNG, H. S. Learning the parts of objects by non-negative matrix factorization. v. 401, p. 788–91, 11 1999. Citado 2 vezes nas páginas 7 e 17.

MISRA, H. et al. Text segmentation via topic modeling: An analytical study. In: *Proceedings of the 18th ACM Conference on Information and Knowledge Management*. New York, NY, USA: ACM, 2009. (CIKM '09), p. 1553–1556. ISBN 978-1-60558-512-3. Disponível em: <http://doi.acm.org/10.1145/1645953.1646170>. Citado 2 vezes nas páginas 9 e 23.

NAILI, M.; CHAIBI, A. H.; GHEZALA, H. H. B. Exogenous approach to improve topic segmentation. *International Journal of Intelligent Computing and Cybernetics*, v. 9, n. 2, p. 165–178, 2016. Disponível em: <https://doi.org/10.1108/IJICC-01-2016-0001>. Citado 2 vezes nas páginas 9 e 24.

PEVZNER, L.; HEARST, M. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, v. 28, n. 1, p. 19–36, 2002. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-0037870455&doi=10.1162%2f089120102317341756&partnerID=40&md5=279abc4e76fcfc2c4a1896e76a245034>. Citado na página 14.

REYNAR, J. C. *Topic Segmentation: Algorithms and Applications*. Tese (Doutorado), Philadelphia, PA, USA, 1998. AAI9829978. Citado na página 10.

REZENDE, S. O. *Sistemas Inteligentes*. Barueri, SP: Manole, 2003. 337 - 270 p. Citado 4 vezes nas páginas 15, 16, 23 e 28.

SAKAHARA, M.; OKADA, S.; NITTA, K. Domain-independent unsupervised text segmentation for data management. In: *2014 IEEE International Conference on Data Mining Workshop*. [S.l.: s.n.], 2014. p. 481–487. ISSN 2375-9232. Citado na página 23.

STEYVERS, M.; GRIFFITHS, T. Probabilistic topic models. In: LANDAUER, T.; MCNAMARA, S. D.; KINTSCH, W. (Ed.). *Latent Semantic Analysis: A Road to Meaning*. Laurence Erlbaum, 2007. cap. Probabilistic topic models. Disponível em: <http://psiexp.ss.uci.edu/research/papers/SteyversGriffithsLSABookFormatted.pdf>. Citado na página 17.