LUCAS FERNANDES BRUNIALTI

Biclusterização aplicada em Sistemas de Recomendação baseados em Conteúdo Textual

São Paulo

2014

LUCAS FERNANDES BRUNIALTI

Biclusterização aplicada em Sistemas de Recomendação baseados em Conteúdo Textual

Texto de Exame de Qualificação apresentado à Escola de Artes, Ciências e Humanidades da Universidade de São Paulo como parte dos requisitos para obtenção do título de Mestre em Ciências pelo Programa de Pós-graduação em Sistemas de Informação.

Orientador: Profa. Dra. Sarajane Marques Peres

Co-Orientador: Prof. Dr. Valdinei Freire da Silva

São Paulo

2014

Autorização para Reprodução

Ficha catalográfica

Folha de Aprovação

Texto de Exame de Qualificação de Mestrado sob o título *"Biclusterização aplicada em Sistemas de Recomendação baseados em Conteúdo Textual"*, apresentado por Lucas Fernandes Brunialti e aprovado em ___ de _____ de _____, em São Paulo, Estado de São Paulo, pela comissão examinadora constituída pelos doutores:

Prof. Dr. _____
Presidente
Instituição: _____

Prof. Dr. _____
Instituição: _____

Prof. Dr. _____
Instituição: _____

Resumo

BRUNIALTI, Lucas Fernandes. **Biclusterização aplicada em Sistemas de Recomendação baseados em Conteúdo Textual**. 2015. NúmeroDePáginas f. Dissertação (Mestrado em Ciências) – Escola de Artes, Ciências e Humanidades, Universidade de São Paulo, São Paulo, Ano2.

Biclusterização representa uma estratégia de análise de dados com potencial para encontrar grupos de objetos similares considerando a correlação parcial existente entre os atributos descritivos dos mesmos. Esse potencial pode ser particularmente útil para Sistemas de Recomendação baseados em conteúdo, nos quais se faz necessária a sugestão de itens úteis, porém diversificados, para um usuário. Esta necessidade configura-se como um problema de serendipidade, uma das propriedades de Sistemas de Recomendação. O objetivo deste trabalho é analisar a aderência dos resultados provenientes de algoritmos de biclusterização, aplicados aos itens a serem recomendados, à meta de otimização da serendipidade. Para alcançar tal objetivo, este trabalho propõe um estudo da aplicação de algoritmos clássicos de biclusterização à conteúdo textual referente ao escopo de um Sistema de Recomendação de notícias. Ainda, este trabalho propõe a utilização de *essembles* como uma forma alternativa de encontrar biclusters. A hipótese defendida é que devido à análise de correlações parciais dos atributos descritivos dos dados, seria possível encontrar notícias com similaridades parciais entre si devido às particularidades presentes nas mesmas. Desta forma, uma mesma notícia que cobre mais de um contexto poderia ser recomendada a usuários que estariam, à princípio, interessandos em assuntos diferentes. Como forma de avaliação dos resultados obtidos, é proposta uma análise comparativa entre os resultados da recomendação obtida via biclusterização e os resultados de recomendação obtida via filtro colaborativo, onde o problema da serendipidade é reconhecidamente bem resolvido.

Palavras-chave: Biclusterização. Sistemas de Recomendação. Recomendação de Notícias. Mineração de Textos. *Essembles*.

Abstract

BRUNIALTI, Lucas Fernandes. **Work title**. 2015. NumberOfPages p. Dissertation (Master of Science) – School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, Year2.

Write here the English version of your "Resumo "...

Keywords: Keyword1. Keyword2. Keyword3. Etc.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Sugestão: fazer uma introdução contextualizada em um probema de recomendação de notícias. Para esse problema, criar um cenário onde uma notícia poderia ser recomendada para dois usuários diferentes que estariam inicialmente navegando em notícias diferentes (de contextos diferentes). Mostrar que uma notícia pode estar relacionada a duas outras que por sua vez não estão relacionadas.

Texto texto.

Texto texto.

Texto texto.

1.1 Apresentação do Problema de Recomendação

Apresentar de maneira mais formal (acima foi só um cenário), o problema de recomendação. Tentar caracterizar (aí acho que de maneira não formal necessariamente)

a propriedade de serendipidade.

Texto texto.

1.2 Apresentação do contexto de Biclusterização

Apresentar o que é biclusterização, em linhas gerais e com uma pequena formulação. Falar que existem vários algoritmos.

Texto texto.

Texto texto.

Apresentar em linhas gerais a possibilidade de implementar bicluterização usando a alternativa de essemble.

Texto texto.

Texto texto texto texto texto texto texto texto texto texto texto texto texto texto

texto texto.

1.3 Hipótese

Um grande problema referente à SRs baseados em conteúdo é a sobre-especialização, que é a falta de recomendações com conteúdo novo ou inesperado à um usuário, ou a falta de serendipidade nas recomendações. Como a Biclusterização é capaz de expandir o espaço de busca, levando em consideração subconjuntos das características para a formação de clusters, a hipótese desse projeto é que a aplicação de algoritmos de biclusterização em SRs baseados em conteúdo pode amenizar o problema de sobre-especialização, melhorando a qualidade das recomendações e contribuindo para a área de SRs.

1.4 Objetivos

O objetivo geral desse trabalho é amenizar o problema da serendipidade em SRs baseados em conteúdo textual, no domínio de notícias. Isso será possível através da aplicação de algoritmos de biclusterização nessas notícias, fazendo uso de diversas representações para estruturação do conteúdo textual.

Assim, foram estabelecidos os seguintes objetivos específicos:

- levantamento do referencial teórico de sistemas de recomendação textual
    - levantamento do estado da arte (sua revisao sistemática)
- levantamento de referencial teórico de biclustering
- construção de um conjunto de dados
- estudo de processamento de texto (text mining)
- estudo das técnicas de biclustering aplicadas a texto
- estudo das técnicas de essemble aplicadas a text

1.5 Metodologia

Aqui explicar cada um dos passos. Explicar que o levantamento do referencial teórico através de análise exploratória. Explicar como é a revisão sistemática (um parágrafo). Explicar a motivação de tê-la realizado. Texto texto.

Explicar como o conjunto de dados foi construído. Explicar a base de cliques. Texto texto.

Explicar que as técnica de biclustering e essemble serão aplicadas aos textos processados e que uma avaliação será feita com base nas informações de cliques. Texto texto.

1.6 Organização do documento

Este documento é composto por X capítulos incluindo esta introdução. O Capítulo 1 ..... e apresentar cada um. Texto texto.

Capítulo 2

Conceitos Fundamentais

Este capítulo introduz os conceitos fundamentais para o entendimento dessa dissertação, fazendo um apanhado dos conceitos na área de Sistemas de Recomendação (seção 2.1), Biclusterização (seção **??**) e Mineração de Texto (seção **??**).

2.1 Sistemas de Recomendação

A grande maioria das pessoas que usam a Internet muito provavelmente já interagiram com algum Sistema de Recomendação (SR), por isso, o seu conceito é intuitivo. Porém, por ser uma área relativamente nova (LOPS; GEMMIS; SEMERARO, 2011), muitos autores não usam uma definicão que reflete a realidade dos SRs, isso acontece por ser uma área relativamente nova que teve um crescimento muito grande nos últimos anos (tese-chap2).

Resnick e Varian (1997), quem criou o termo Sistemas de Recomendação (tese-chap2) (NEUMANN, 2007), argumentam que Sistemas de Recomendação servem para nos ajudar em processos de tomada de decisão do nosso dia-a-dia, como quais itens comprar, quais músicas ouvir, ou quais notícias ler. Além disso, Resnick e Varian (1997) provê uma taxonomia para definição de Sistemas de Recomendação:

- Conteúdo recomendado: os itens que são recomendados pelo Sistema de Recomendação, ex: produtos, músicas, notícias e/ou etc.
- Entrada dos usuários: as interações que os usuários realizam com os itens são a entrada para um SR, estas podem ser implícitas (ex: o usuário x leu a notícia y) ou explícitas (ex: o usuário x classificou o filme y como 5 estrelas).
- Target of recommendation: os itens recomendados podem ser diretamente para um usuário (personalizado), direcionados para um grupo de usuários ou todos os usuários (não-personalizado).
- Técnicas para recomendação (agregações): qual as estratégias e os algoritmos que os SRs usam para criar recomendações.

- Uso das recomendações: trata-se de como mostrar as recomendações para os usuários, ex: filtrando recomendações negativas, ordenando pelo fator numérico, etc.

Porém, definições mais recentes (BURKE, 2002, 2007), descrevem SRs como qualquer sistema que produz recomendações personalizadas ou tem o efeito de guiar um usuário de modo personalizado, mostrando itens que possam ser interessantes para este usuário, dentro de uma grande quantidade de opções. Isso faz com que SRs que provêm recomendações não personalizadas deixem de adequar com a definição de SR. É provável que isso se deve ao fato que das estratégias usadas atualmente, que têm o foco de produzir recomendações personalizadas.

Formalmente, Burke (2002, 2007) define SRs como um conjunto de itens I que podem ser recomendados e um conjunto de usuários U que as preferências são conhecidas, um usuário u pra o qual as recomendações são geradas, e algum item i que queremos predizer a preferência para u. Adomavicius e Tuzhilin (2005) extende a definição com uma função de utilidade f que mede o quão útil é o item i para o usuário u: $f : I \times U \rightarrow R$, em que $R = \{r_{u_1,i_1}, r_{u_1,i_2}, \ldots, r_{u_1,i_m}, \ldots, r_{u_1,i_m}\}$ é um conjunto ordenado com valores faltantes, sendo r_{u_i,i_j} um inteiro ou real que representa a interação do usuário u_i no item i_j. No entanto, existem tipos de SRs que não estimam f completamente, podendo otimizar funções auxiliares para gerar as recomendações para um usuário u (LOPS; GEMMIS; SEMERARO, 2011).

Em síntese, um Sistema de Recomendação tem a função de auxiliar os usuários de uma aplicação à interagir com itens, provendo sugestões de quais itens interagir, baseando-se no histórico de interações desses usuários com esses itens.

2.1.1 Tipos de Sistemas de Recomendação

Para estimar f e chegar no conjunto ordenado R existem diversas estratégias, daí surgem os tipos de SRs. Os tipos de SRs diferem quanto ao domínio, informações usadas para recomendação, algoritmos (FELDMAN; SANGER, 2006), e principalmente nas propriedades em que cada tipo se destaca.

Burke (2002, 2007) provê uma taxonomia já considerada clássica Lops, Gemmis e Semeraro (2011), que categoriza os SRs em cinco diferentes tipos:

- *Filtragem colaborativo*, o primeiro tipo de SR que foi implementado (RESNICK; VARIAN, 1997), tem como idéia básica encontrar outros usuários $u_{1,\ldots,n}$ em U, sendo

$n < |U|$ e não necessariamente em ordem, com preferências semelhantes à u, e então recomendar itens que $u_{1,...,n}$ interagiram e que u ainda não interagiu, estabelecendo alguma métrica para estimar f. Medidas geralmente usadas incluem *Correlação de Pearson* e *Similaridade dos Cossenos*, também são usados técnicas para redução de dimensionalidade, como *Decomposição de Valores Singulares* e *Fatorização de Matriz* (JANNACH et al., 2011).

- *Baseado em conteúdo*, é um dos tipos de SR que otimiza funções auxiliares à f. Descreve os itens por características possibilitando o uso de medidas de similaridade entre itens. Então, com as interações dos usuários de U sob itens em I, é construído um perfil de interesses para cada usuário. As recomendações são feitas a partir da combinação do perfil de interesses de um usuário u com os itens em I que u ainda não interagiu. Neste caso são usadas técnicas de Recuperação de Informação (JANNACH et al., 2011) para representar os itens e calcular similaridades entre itens, assim como técnicas de Aprendizado de Máquina Supervisionado e não-supervisionado (JANNACH et al., 2011; BURKE, 2002).

- *Baseado em Conhecimento*, tem o intuito de sugerir itens, de forma personalizada, baseando-se nas necessidades ou regras estabelecidas por um usuário u e nas características dos itens em I. São estabelecidas medidas de similaridade para estimar o quanto as necessidades do usuário match as recomendações (JANNACH et al., 2011; LOPS; GEMMIS; SEMERARO, 2011; BURKE, 2007).

- *Híbrido*, é capaz de combinar as vantagens de cada tipo de SR descrito para suprir as limitações associadas à cada tipo. A dificuldade esta em como combinar as diferentes técnicas de cada algoritmo (JANNACH et al., 2011; BURKE, 2007). Burke (2007) identificou 7 tipos de SRs Híbridos em uma revisão da literatura: Pesagem, atribui um peso para cada algoritmo; Switching, seleciona um dos algoritmos (ou tipos); Mixed, recomendações são mostradas em conjunto; Combinação de características, diferentes fontes são combinadas em apenas um algoritmo; Feature Augumentation, uma técnica é usada para computar características que servem de entrada para outra técnica; Cascade, é atribuído um grau de prioridade para cada algoritmo; Meta-level, uma técnica gera um modelo, que é usado como entrada para outras técnicas.

2.1.1.1 Vantagens e desvantagens

Cada um dos tipos de SRs descritos possuem algumas limitações independentes se comparados com os outros tipos ou não. Em Jannach et al. (2011), Adomavicius e Tuzhilin (2005), Burke (2002), Lops, Gemmis e Semeraro (2011) são nomeadas os problemas no desenvolvimento de SRs de *novo usuário* (*user cold-start*), *novo item* (*item cold-start*), *esparsidade*, *sobre-especialização* e *análise de conteúdo limitada* referentes aos SRs.

Os problemas de *novo usuário* e *novo item* são similares, basicamente, enquanto o primeiro se trata da dificuldade de gerar recomendações para novos usuários, o segundo se trata de gerar recomendações para novos itens. O problema de *novo usuário* esta presente como uma desvantagem nos SRs de filtragem colaborativa e baseado em conteúdo, pois o SR não conhece as preferências dos novos usuários, tendo dificuldade de construir um modelo que tem como base essas preferências. Já o problema de *novo item* é considerado uma vantagem para os SRs baseados em conteúdo, enquanto uma desvantagem para os SRs baseados em filtragem colaborativa. Como a filtragem colaborativa se baseia apenas nas interações de usuários em itens, um item novo, que não teve ou teve poucas interações, não será recomendado, diferentemente do SR baseado em conteúdo, que leva em consideração a representação do item para construir recomendações.

Contrariamente, o problema de *sobre-especialização* é uma vantagem para os SRs de filtragem colaborativa e uma desvantagem para os SRs baseados em conteúdo. A *sobre-especialização* diz respeito ao problema de sugerir apenas itens previsíveis para o usuário, por exemplo, se o usuário viu notícias apenas de esporte, ele já espera receber sugestões de notícias de esporte, porém este usuário muito provavelmente pode gostar de ler notícias de outras categorias. A capacidade do SR de sugerir notícias imprevisíveis é chamado de *serendipidade* (JANNACH et al., 2011; LOPS; GEMMIS; SEMERARO, 2011). SRs baseados em conteúdo sofrem desse problema pois combina o perfil de preferências de um usuário com os itens em I, restrigindo o espaço de busca para realizar a sugestão de itens. Enquanto isso, os SRs baseados em filtragem colaborativa são capazes de oferecer sugestões úteis e serendipitas, pois são capazes de ampliar o espaço de busca através da estratégia de sugerir itens que usuários semelhantes à u interagiram e que u ainda não interagiu.

Os SRs baseados em filtragem colaborativa são os únicos que sofrem do problema de *esparsidade*, que é o fato de usuários interagirem com apenas um pequeno subconjunto do conjunto de itens, tornando a matriz de interações ou preferências ($U \times I$) esparsa. Isso faz com que aumente a necessidade de ter uma grande quantidade de usuários, pois

este tipo de SR necessita de intersecções nas interações de itens por usuários, para que seja possível encontrar usuários similares a um dado usuário.

Assim como os SRs de filtragem colaborativa, os SRs baseados em conteúdo sofrem de um problema único, que é a *análise de conteúdo limitada*. Este problema se refere à representação dos itens, que tem de ser suficiente para discriminá-los (LOPS; GEMMIS; SEMERARO, 2011). Em SRs baseados em conteúdo é comum ter que limitar a representação de itens, tanto pelo número de características quanto pela modelagem. Por exemplo, no contexto de notícias na web, se a representação for um vetor com o número de ocorrências de cada palavra, perdemos a relação entre as palavras, e também todo o conteúdo que não é texto, como imagens, vídeos, etc.

2.1.2 Avaliação da Recomendação

Diferentes técnicas foram mostradas e uma variedade de problemas relacionados à essas técnicas, mas como avaliar se uma estratégia adotada no desenvolvimento realmente é efetiva? SRs podem ser avaliados através de *experimentos online*, que podem descobrir a real influência do SR no comportamento do usuário, e *experimentos offline*, que estimam o erro de predição e simulam o comportamento do usuário no SR usando um conjunto de dados (SHANI; GUNAWARDANA, 2011).

Para experimentos online podem ser estabelecidas variáveis através da captura implícita ou explícita do comportamento, como satisfação do usuário e taxa de cliques (*click-through rate* - CTR). Uma das maneiras para realizar esse tipo de experimento é por meio de testes A/B (JANNACH et al., 2011), em que cada usuário, ao interagir com o SR, recebe um tratamento diferente aleatóriamente, dentro dos possíveis tratamentos estabelecidos pelo experimento. Assim, é possível dizer, por exemplo, a influência de um novo componente no SR no comportamento dos usuários.

Tradicionalmente, SRs são avaliados através de experimentos offline (JANNACH et al., 2011). Por serem simples, esses tipos de experimentos podem ser usados para para selecionar algoritmos, no entanto, Shani e Gunawardana (2011) argumentam que não é possível medir diretamente a influência das recomendações no comportamento dos usuários. Para simular o comportamento do usuário em um SR usando um conjunto de dados, são estabelecidos dois subconjunto das interações de u aleatoriamente ($\{r_{u,i_4}, r_{u,i_3}, \dots\}$), um para treinamento dos modelos conj treino$_u$ e outro para teste conj teste$_u$. Assim é possível adotar estratégias semelhantemente com às adotadas em Aprendizado de Máquina, como *matriz de confusão*, *precisão*, *revocação*, *f1-score*, *cross-validation* e etc.

A técnica de *matriz de confusão* é possível ser usada em *experimentos online* e *offline*, da seguinte maneira: se o usuário gostar do item sugerido à ele, é considerada uma predição correta (verdadeiro positivo); se o usuário não solicita preferência pela sugestão ou se não existe informações da preferência do usuário para esta sugestão, será considerada uma predição errada (falso positivo); contrariamente, se o SR não fazer essas sugestões, é considerada uma emissão correta (verdadeiro negativo); por fim, se o SR não sugerir itens que o usuário tem preferência, é consuderada uma predição errada (falso negativo).

Uma das métricas mais comuns usadas para *experimentos offline* é o *erro absoluto médio* (*Mean Absolute Error* - MAE) e *raiz do erro quadrático médio* (*Root Mean Squared Error* - RMSE) (JANNACH et al., 2011), o qual foi usado como métrica para a competição Netflix Prize[1], que teve grande repercursão na academia e na indústria. A defição formal das métricas MAE e RMSE são descritas nas equações 2.1 e 2.2, respectivamente.

$$MAE = \sum_{u \in U} \frac{\sum_{i \in \text{conj teste}_u} |f(u,i) - r_{u,i}|}{|\text{conj teste}_u|} \tag{2.1}$$

$$RMSE = \sum_{u \in U} \sqrt{\frac{\sum_{i \in \text{conj teste}_u} (f(u,i) - r_{u,i}^2}{|\text{conj teste}_u|}} \tag{2.2}$$

A métrica MAE computa o erro médio entre as predições feitas pelo SR ($f(u,i)$) e os valores reais das preferências dos usuários ($r_{u,i}$) para todos os usuários em U, enquanto o RMSE amplifica erros grandes, pois eleva o mesmo ao quadrado. Essas métricas são usadas para valores reais, ou seja, $r_{u,i} \in [0,1]$, por exemplo.

Para valores binários de $r_{u,i}$ ou quando deseja-se prever o número de recomendações relevantes para um usuário u, são usadas as métricas *precisão* e *revocação* (JANNACH et al., 2011).

Para medir a serendipidade das recomendações, Shani e Gunawardana (2011) propõem uma estratégia: estabelecer uma medida de distância entre itens e rotular os itens com menor distância entre si como ausentes de serendipidade, assim, algoritmos que evitem esses itens, serão considerados superiores.

[1] http://www.netflixprize.com/

2.2 Biclusterização

Técnicas e algoritmos de biclusterização são usadas, principalmente, no contexto de expressão genética. No entanto, algoritmos de biclusterização se fazem útil quando se deseja encontrar *modelos locais*. Ou seja, enquanto algoritmos de clusterização têm o intuito de encontrar *modelos globais*, que geram grupos de dados levando em consideração todas as características, algoritmos de biclusterização geram grupos de dados em que as características tem alta correlação (MADEIRA; OLIVEIRA, 2004).

Para a descrição do problema formal de biclusterização usa-se a seguinte definição (MADEIRA; OLIVEIRA, 2004): uma matriz A, $n \times m$, um conjunto de linhas $X = \{x_1, \ldots, x_n\}$ e um conjunto de colunas $Y = \{y_1, \ldots, y_m\}$, em que a_{ij}, geralmente um número real, e representa a relação entre a linha x_i e a coluna y_j. O problema de biclusterização é encontrar biclusters, que são submatrizes de A, denotados por A_{IJ}, em que $I \subseteq X$ e $J \subseteq Y$. Assim, o bicluster A_{IJ} é um grupo dos exemplos em I, perante as características com alta correlação J.

2.2.1 Tipos de biclusters

Como a definição de bicluster não inclui uma prévia estrutura da matriz A e dos biclusters A_{IJ}, diversos algoritmos propostos na literatura diferem quanto ao tipo de bicluster que são capazes de encontrar. Uma taxonomia dos tipos de biclusters é proposta por Madeira e Oliveira (2004):

- *Biclusters com valores constantes*, se trata de biclusters em que todos os valores de A_{IJ} são constantes: $a_{ij} = \mu, \forall i, j \in I, J$, onde μ é um valor constante dentro de A_{IJ}. Porém, em conjuntos de dados reais, esses biclusters estão presentes com algum tipo de ruído $\mu + \eta_{ij}$, onde η_{ij} é o ruído associado com os valures de μ e a_{ij} (MADEIRA; OLIVEIRA, 2004).

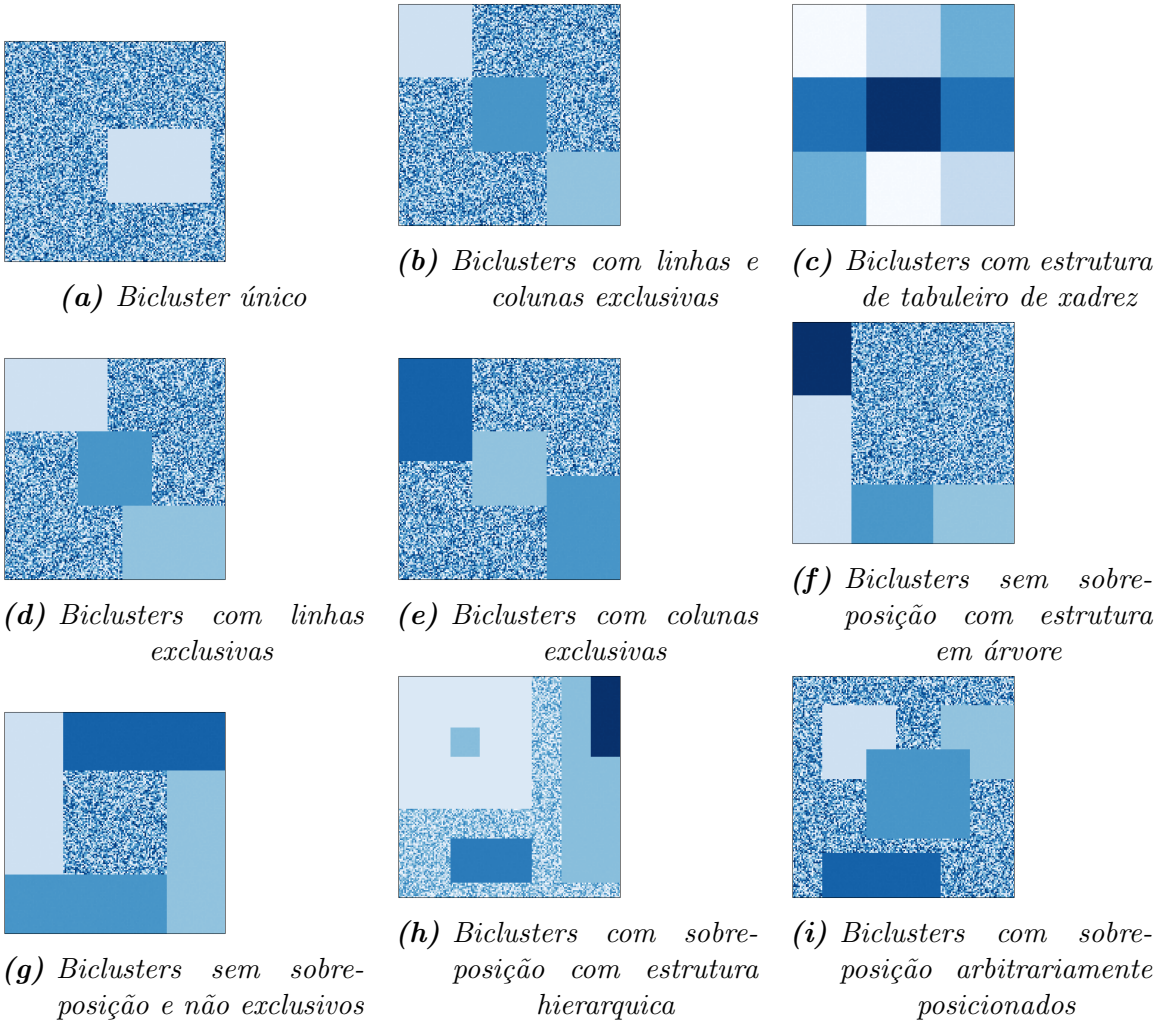
- *Biclusters com valores constantes nas linhas ou colunas*, se trata de biclusters com valores constantes nas linhas: $a_{ij} = \mu + \alpha_i, \forall i, j \in I, J$ ou $a_{ij} = \mu \cdot \alpha_i, \forall i, j \in I, J$, onde α_i é um fator aditivo ou multiplicativo para cada linha; ou ainda biclusters com valores constantes nas colunas: $a_{ij} = \mu + \beta_j, \forall i, j \in I, J$ ou $a_{ij} = \mu \cdot \beta_j, \forall i, j \in I, J$, onde β_j é um fator aditivo ou multiplicativo para cada coluna (MADEIRA; OLIVEIRA, 2004).

- *Biclusters com valores coerentes*, em que são considerados valores próximos entre

si (coerentes) para definição de um bicluster: $a_{ij} = \mu + \alpha_i + \beta_j, \forall i, j \in I, J$, ou $a_{ij} = \mu' \cdot \alpha'_i \cdot \beta'_j, \forall i, j \in I, J$, sendo que se $\mu = \log \mu' \implies \alpha_i = \alpha'_i, \beta_j = \beta'_j$ (MADEIRA; OLIVEIRA, 2004).

- *Biclusters com evoluções coerentes*, têm seus valores com evoluções coerentes, por exemplo, um bicluster com $a_{i4} \leq a_{i3} \leq a_{i2} \leq a_{i1}$ tem valores com evolução coerente na coluna (MADEIRA; OLIVEIRA, 2004). Seus valores podem ser gerados por uma função geradora de valores com evolução coerente $a_{ij} = g(a_{ij}), \forall i, j \in I, J$, sendo $g(\cdot)$ não linear e não constante, para que o tipo de bicluster não seja classificado nos casos anteriores.

Os biclusters também se diferem quanto a sua estrutura, cada algoritmo faz uma suposição da estrutura de biclusters que é capaz de encontrar. A Figura 1 sumariza as diferentes estruturas de biclusters, com as linhas e colunas ordenadas, para permitir a visualização dos biclusters por meio do mapa de calor dos valores de A.



(a) Bicluster único

(b) Biclusters com linhas e colunas exclusivas

(c) Biclusters com estrutura de tabuleiro de xadrez

(d) Biclusters com linhas exclusivas

(e) Biclusters com colunas exclusivas

(f) Biclusters sem sobreposição com estrutura em árvore

(g) Biclusters sem sobreposição e não exclusivos

(h) Biclusters com sobreposição com estrutura hierarquica

(i) Biclusters com sobreposição arbitrariamente posicionados

Figura 1 – *Diferentes estruturas de biclusters, quadrados com cores sólidas representam biclusters (Adaptado de (MADEIRA; OLIVEIRA, 2004))*

Texto texto.

Texto texto.

2.2.2 Algoritmos para biclustering

Texto texto.

Texto texto.

Texto texto.

2.2.3 Avaliação de biclustering

Texto texto.

Texto texto.

2.3 Mineração de Texto

Técnicas de Mineração de Texto são muito usadas para SRs baseados em conteúdo textual (LOPS; GEMMIS; SEMERARO, 2011), principalmente quando o contexto do SR se trata de informações não-estruturadas. Mineração de Texto lida com análise de texto, suportando a sua natureza não-estruturada, imprecisa, incerta e difusa, para extração de informação e conhecimento (HOTHO; NüRNBERGER; PAAß, 2005). Além disso, a área de Mineração de Texto utiliza de técnicas das áreas de Recuperação de Informação, Processamento de Linguagem Natural (PLN), conectando essas técnicas com algoritmos e métodos de Descoberta de Conhecimento em Banco de Dados, Mineração de Dados, Aprendizado de Máquina e Estatística (HOTHO; NüRNBERGER; PAAß, 2005).

Feldman e Sanger (2006), apresentam uma arquitetura geral para aplicações de Mineração de Textos composta por quatro etapas: tarefas de pré-processamento, que preparam os dados para a central de operações de mineração; central de operações de mineração, incluem algoritmos para a descoberta de padrões, tendências e conhecimentos através de técnicas e algoritmos; componentes de apresentação, incluem interfaces para o usuário apresentando visualizações dos conhecimentos gerados na etapa anterior; e técnicas de refinamento, também descrito como uma fase de pós-processamento, que inclui métodos para filtrar informações redundantes.

2.3.1 Tarefas de pré-processamento

As tarefas de pré-processamento incluem rotinas, processos e métodos para a estruturação dos textos presentes nos documentos. A estruturação se faz necessária para a extração de informações e descoberta de conhecimento por meio de técnicas e algoritmos (HOTHO; NüRNBERGER; PAAß, 2005).

2.3.1.1 Representação textual

Para a estruturação dos textos é necessário a definição da representação textual dos documentos. O vetor de palavras, ou Vector Space Model (SALTON; WONG; YANG, 1975), é a representação clássica usada para representar documentos textuais (referencia). Cada dimensão desse vetor esta associado à palavra, sendo que todas as dimensões representam todas as palavras do conjunto de documentos. Formalmente, temos um conjunto de documentos $D = \{d_1, d_2, \ldots, d_n\}$, em que d_i representa um documento e n o número total de documentos, e um conjunto de palavras $P = \{p_1, p_2, \ldots, p_m\}$, em que

p_j representa uma palavra e m o número de palavras presentes em todos os documentos. Representando a frequência de uma palavra, o número de vezes que p_j aparece em um documento d_i, por $fp(p_j, d_i)$, o vetor de palavras pode ser construído e representado da seguinte forma: $\vec{vp}_{d_i} = (fp(p_1, d_i), fp(p_2, d_i), \ldots, fp(p_m, d_i))$. Salton, Wong e Yang (1975) argumenta que a representação textual de documentos em vetor de palavras é suficiente para separar documentos. Ao invés de frequência de palavra, também é comumente usado, a representação binária (referencia), ou seja, p_j aparecendo em d_i corresponde à entrada 1 na dimensão j em $\vec{vp}_{d_i}$. Há também outros métodos para representação textual, como *n-gramas* e *ontologias*.

Ainda sobre o vetor de palavras, Salton, Wong e Yang (1975) mostra com experimentos em diversos conjuntos de dados, que o uso da normalização nos vetores usando a técnica de Frequência de Palavras-Frequência de Documentos Inversa (*Term Frequency-Inversed Document Frequency* - TFIDF) é capaz de melhorar a separação de documentos:

$$\begin{aligned} \textit{fp-fdi}(p_j, d_i) &= fp(p_j, d_i) \cdot fdi(p_j) \\ \textit{fp-fdi}(p_j, d_i) &= fp(p_j, d_i) \cdot \left(log_2 \frac{n}{fd(p_j) + 1} \right) \end{aligned} \tag{2.3}$$

em que $fdi(p_j)$ representa a Frequência de Documentos Inversa da palavra p_j e $fd(p_j)$ a frequência de documentos que contém p_j. Essa normalização faz com que a frequência das palavras que aparecem em muitos documentos seja reduzida, e a frequência das palavras que aparecem em alguns raros documentos seja aumentada, com um fator de $\log_2$.

2.3.1.2 Tokenização

Para realizar a estruturação de textos e representar os textos dos documentos em vetores de palavras, o primeiro processo a ser realizado é a *tokenização*, que cria um dicionário de palavras para cada documento através da quebra dos textos desses documentos. A quebra do texto pode ser feita através de caracteres delimitadores de termos, como espaços em branco, pontuações e etc. No entanto, existem casos que esses caracteres podem não ser delimitadores de termos, como por exemplo os termos *Prof.* e *Sr.*. Este problema é chamado de determinação de fim de sentença, e pode ser resolvido por métodos estáticos (*hard-coded*), baseados em regras e métodos de Aprendizado de Máquina (WEISS; INDURKHYA; ZHANG, 2010).

2.3.1.3 Filtragem

Métodos de filtragem têm a função de retirar palavras do conjunto P que não contribuem para distinguir ou identificar documentos, como exemplo, conjunções (*e*, *pois*, *que*), artigos (*um*, *o*, *a*), preposições (*de*, *para*) e etc. A técnica de retirar determinadas palavras de P a partir de uma lista, é chamada de *stopwords*. Também são usadas outras técnicas, como a eliminação de palavras com a frequência muito alta ou muito baixa.

2.3.1.4 Stemming

A fim de reduzir a ambiguidade de palavras, o método de *stemming* é capaz de juntar em uma única forma palavras relacionadas (MINER et al., 2012), como exemplo o verbo *fazer*, que pode se apresentar em diversas formas, como *fazendo*, *fez*, etc. Esse processo pode ser capaz de aumentar a capacidade da representação em distinguir ou identificar documentos.

2.3.1.5 Redução de Dimensionalidade

A representação em vetor de palavras pode resultar em vetores esparsos num espaço de alta dimensão, que pode fazer com que algoritmos sofram do problema de *Maldição de Dimensionalidade* (footnote haykin). Para amenização desse problema, são usados métodos de *redução de dimensionalidade*. A técnica mais comum de *redução de dimensionalidade* é chamada *Análise dos Componentes Principais* (*Principal Component Analysis - PCA*) (MURPHY, 2012). Esta técnica tem o objetivo de encontrar uma representação compacta através da descoberta de k vetores n-dimensionais ortogonais aos dados ($\vec{v}$), em que $k \leq m$. Os vetores são encontrados a partir da minimização da projeção dos dados em $\vec{v}$. Depois de encontrados os vetores $\vec{v}$, é feita a projeção dos dados nesses vetores, resultando em uma representação num espaço mais compacto (HAN; KAMBER; PEI, 2011). É possível aplicar o algoritmo *PCA*, no vetor de palavras, diminuindo a dimensionalidade e esparsidade, superando o problema de *Maldição de Dimensionalidade*.

Capítulo 3

Sistemas de Recomendação por Conte e Aprendizado de Máquina

Texto texto.

Texto texto.

Texto texto.

Texto texto.

Texto texto

texto texto.

Texto texto.

Texto texto.

Capítulo 4

Proposta

Texto texto.

4.1 Apresentação do corpus IG

O corpus do IG é composto por um conjunto de notícias $\{n_1, n_2, \ldots, n_m\}$ em que cada notícia n_i é representado pela tupla (t, st, b, c_i), onde t é o título, st o subtítulo, b o corpo da notícia e c_i um elemento do conjunto de canais $\{gente, ultimosegundo, delas, economia, esporte, saude, igay, deles, tecnologia, receitas, igirl, jovem, arena, luxo\}$, em que cada canal representa um assunto ou tópico de notícias.

O número total de notícias é $m = xxx$, ...

4.1.1 Pré processamento do corpus IG

Texto texto.

Texto texto

texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto texto.

4.2 Estudos iniciais de biclustering no corpus IG

Texto texto.

Texto texto.

Texto texto.

Texto texto.

4.3 Próximos passos - cronograma

Texto texto

texto texto.

Texto texto.

Texto texto.

Referências

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

BURKE, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, Kluwer Academic Publishers, Hingham, MA, USA, v. 12, n. 4, p. 331–370, 2002. ISSN 0924-1868.

BURKE, R. Hybrid web recommender systems. In: *The Adaptive Web: Methods and Strategies of Web Personalization*. Berlin: [s.n.], 2007. cap. 12, p. 377–408.

FELDMAN, R.; SANGER, J. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge, MA, USA: Cambridge University Press, 2006. Hardcover. ISBN 0521836573.

HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN 0123814790, 9780123814791.

HOTHO, A.; NüRNBERGER, A.; PAAß, G. A brief survey of text mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology*, v. 20, n. 1, p. 19–62, maio 2005. ISSN 0175-1336. Disponível em: <http://www.kde.cs.uni-kassel.de/hotho/pub/2005/hotho05TextMining.pdf>.

JANNACH, D. et al. *Recommender Systems An Introduction*. [S.l.]: Cambridge University Press, 2011.

LOPS, P.; GEMMIS, M. de; SEMERARO, G. Content-based recommender systems: State of the art and trends. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. [S.l.]: Springer, 2011. p. 73–105. ISBN 978-0-387-85819-7.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 1, p. 24–45, January 2004. ISSN 1545-5963.

MINER, G. et al. *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*. 1st. ed. [S.l.]: Academic Press, 2012. ISBN 012386979X, 9780123869791.

MURPHY, K. P. *Machine Learning: A Probabilistic Perspective*. [S.l.]: The MIT Press, 2012. ISBN 0262018020, 9780262018029.

NEUMANN, A. Motivating and supporting user interaction with recommender systems. In: KOVáCS, L.; FUHR, N.; MEGHINI, C. (Ed.). *Research and Advanced Technology for Digital Libraries*. [S.l.]: Springer Berlin Heidelberg, 2007, (Lecture Notes in Computer Science, v. 4675). p. 428–439. ISBN 978-3-540-74850-2.

RESNICK, P.; VARIAN, H. R. Recommender systems. *Communications of the ACM*, ACM, New York, NY, USA, v. 40, p. 56–58, March 1997.

SALTON, G.; WONG, A.; YANG, C. S. A vector space model for automatic indexing. *Communications of the ACM*, ACM, New York, NY, USA, v. 18, n. 11, p. 613–620, 1975. ISSN 0001-0782.

SHANI, G.; GUNAWARDANA, A. Evaluating recommendation systems. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. [S.l.]: Springer, 2011. p. 257–297. ISBN 978-0-387-85819-7.

WEISS, S. M.; INDURKHYA, N.; ZHANG, T. *Fundamentals of predictive text mining*. London; New York: Springer-Verlag, 2010. ISBN 1849962251 9781849962254 9781849962261 184996226X.