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

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 clusters 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. 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 pertence a mais de um contexto poderia ser recomendada à usuários que estariam, à princípio, interessados 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 Texto.

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, 2015.

Biclustering represents a strategy for Data Analysis with potential to find clusters of similar objects considering partial correlation between their descriptive attributes. This potential can be especially useful for Content Based Recommender Systems, in which the suggestion of useful items is necessary, however diverse, for a user. This need is configured as a serendipity problem, one of the properties of Recommender Systems. The objective of this study is to analyze the adherence of results derived from biclustering algorithms, applied to the items to be recommended, aiming to optimize the serendipity property. To achieve this goal, this thesis proposes a study of applied classical biclustering algorithms to textual content regarding the scope of a news Recommendation System. The hypothesis is that due to the partial correlation analysis of the data descriptive attributes by these algorithms, it is possible to find news with partial similarities due to the characteristics of these news. Thus, the same news that belongs to more than one context could be recommended for users who would also be interested in different subjects. In order to evaluate the results obtained, we propose a comparative analysis results obtained from recommendations obtained by biclustering algorithms and recommendations obtained by collaborative filtering, where the problem of serendipity is admittedly well resolved.

Keywords: Biclustering. Recommender Systems. News Recomendation. Text Mining.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Dada a quantidade e dinamicidade de informação presente na web, é comum as pessoas, usuárias da web, se verem diante de um problema de difícil resolução quando querem encontrar alguma informação. Ainda mais do que isso, facilmente as pessoas se veêm em situações onde não conseguem definir exatamente o que querem encontrar. Por exemplo, quando usuários da web querem comprar um produto ou ler uma notícia sobre algum assunto específico, frequentemente não conseguem decidir onde o qual produto específico comprar diante de tantas possibilidades, ou até mesmo, embora tenham a intenção de ler sobre um assunto específico, encontram tanta informação que não entendem mais exatamente o que estão procurando. Ricci, Rokach e Shapira (2011) argumentam que muitos usuários da web não têm conhecimento suficiente para saber o que querem consumir da web, ou o que não querem. Com o objetivo de amenizar esse problema, surgem os Sistemas de Recomendação (SRs) que, com base em preferências do usuário, predizem ou sugerem itens (produtos, artigos, músicas e etc), com potencial de serem de grande valia para tais usuários.

Considerando especificamente a ocorrência desse problema no domínio de notícias, uma solução apresentada pelos portais de notícias da web começam a ver é o oferecimento de conteúdo personalizado. No entanto, aprender as preferências dos usuários e ao mesmo tempo oferecer conteúdo útil não é uma tarefa trivial, principalmente no contexto em questão, em que as preferências dos usuários mudam com muita frequência (LI et al., 2011).

Um SR de notícias simples e hipotético poderia apresentar a seguinte estratégia para elaborar suas recomendações: dado o histórico de notícias visitadas pelo usuário, as notícias recomendadas seriam aquelas que mais se assemelham ao seu histórico. Esse sistema seria especializado em oferecer notícias que o usuário já esta acostumado a ler, e portanto apresentaria a desvantagem de não recomendar nenhum tipo de conteúdo novo para o usuário – ainda que esse conteúdo pudesse ser interessante à ele.

Olhando o exemplo do SR hipotético com mais detalhes, considere o caso de dois usuários. O primeiro apresenta um padrão de leitura, e portanto um padrão de navegação,

sobre notícias referentes à esporte. O segundo apresenta um padrão de leitura relacionado à notícias que permeiam o campo da música. O sistema hipotético, muito provavelmente, recomendaria aos tais usuários apenas notícias de esporte e música, respectivamente. Embora essas recomendações pareçam ser ideais, e sob um aspecto de análise objetivo elas o são, é factível assumir que esses usuários estão recebendo um serviço de recomendação prático, mas talvez menos útil do que poderia ser, por dois motivos principais: esses usuários possuem um padrão de navegação que foi aprendido pelo sistema, e muito provavelmente eles já sabem seguir esse padrão e não precisam das recomendações senão por comodidade e praticidade; esses usuários estão presos a esse padrão de navegação por não saberem, ou não terem tido motivação, para sair desse padrão e acessar notícias interessantes que tangeiam seus interesses de alguma forma.

O contexto discutido leva à discussão de uma característica de recomendação denominada *serendipidade*. Um SR precisa ser capaz também de apresentar recomendações com serendipidade, ou seja, que sejam diversas do que é comumente acessado pelo usuário, sem que essas se distanciem muito dos interesses dele. O contexto de recomendações, no caso de notícias, é complexo o suficiente para apresentar itens de recomendação que embora estejam prioritariamente dizendo respeito a um assunto, possuem informações relevantes a outros também.

Uma melhoria para o sistema hipotético (Figura 1) seria dotá-lo da capacidade de perceber, por exemplo, que notícias sobre esporte podem trazer conteúdo interessante sobre música, recomendando uma notícia ao usuário interessado em música que, originalmente, só seria recomendada ao usuário interessado em esporte. Por exemplo, é sabido que eventos de beisebol – o *superbowl* – possuem uma abertura cultural na qual grandes artistas da música fazem apresentações. Da mesma maneira, notícias inicialmente classificadas como sendo de esporte poderiam interessar a quem gosta de música, desde que apresentem informações atraentes sobre música, por exemplo os eventos de esportes radicais, como tirolesa, que acontecem em eventos de música contemporâneos – *rock in rio*.

Assim, faz-se interessante o desenvolvimento de estratégias que permitam melhorar a característica de serendipidade de SR e, especificamente neste trabalho, estratégias de biclusterização, que permitem a análise de correlações parciais entre os atributos descritivos de dados que comporão um grupo de objetos similares, são consideradas candidatas a suprir essa lacuna (HARTIGAN, 1972; MIRKIN, 1996; MADEIRA; OLIVEIRA, 2004).

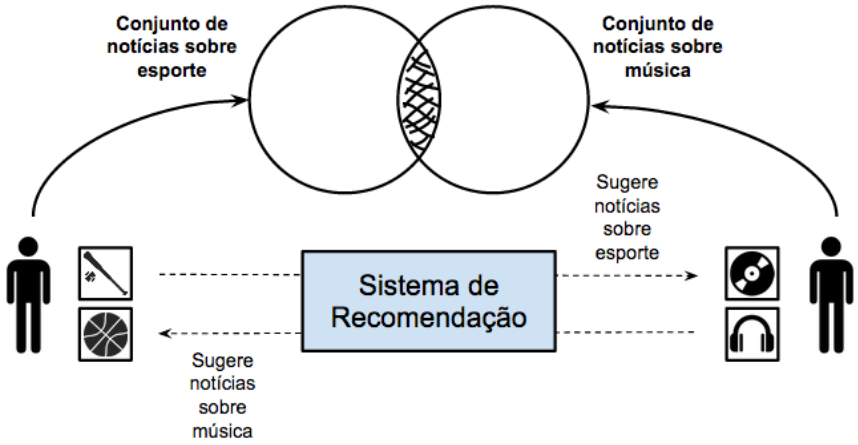


Figura 1 – *Ilustração do caso de dois usuários em um portal de notícias com um SR.*

1.1 Apresentação do Problema de Recomendação

Há diferentes tipos de recomendação que podem ser considerados (Seção 2.1). Aqui, o problema de recomendação com base em conteúdo é apresentado, fazendo uma adaptação de Adomavicius e Tuzhilin (2005).

Seja um conjunto de usuários $U = \{u_1, u_2, \ldots, u_n, \ldots, u_N\}$, um conjunto de itens $\mathcal{I} = \{I_1, I_2, \ldots, I_m, \ldots, I_M\}$ e o histórico $h_{u_n,I_m} \in \mathcal{H}$ indicando se o usuário u_n acessou o item I_m. O problema de recomendação, com base no conteúdo, pode ser visto como a definição de uma função de similaridade entre itens $s : I \to I \times I$ para aproximar uma função $l : \mathcal{H}_u, s \to L_u$, onde $\mathcal{H}_u$ é o subconjunto que representa o histórico de itens que u acessou, e L_u é a lista de recomendações de itens direcionadas à u.

A idéia é que a lista de recomendações L_u permita a ocorrência de *serendipidade*, incluindo um fator de ponderação na função de similaridade s, que considere similaridades parciais.

1.2 Apresentação do contexto de Biclusterização

O problema de biclusterização, em Mineração de Dados, é referente ao processo de clusterização simultâneo de linhas e colunas em um conjunto de dados. Capaz de agrupar objetos similares desse conjunto de dados, considerando seus atributos que apresentem alta correlação, formando grupos de objetos e atributos, simultaneamente, chamados de biclusters. Com maior formalidade, o problema de biclusterização consiste em uma matriz A, de dimensão $N \times M$, um conjunto de linhas (objetos) $X = \{x_1, \ldots, x_N\}$ e um

conjunto de colunas (atributos) $Y = \{y_1, \ldots, y_M\}$, em que a_{nm}, geralmente um número real, e representa a relação entre a linha x_n e a coluna y_m (FRANCA, 2010; MIRKIN, 1996; MADEIRA; OLIVEIRA, 2004).

O problema de biclusterização pode ser ilustrado na Figura 2, onde se tem um conjunto de dados que possui objetos x_1, x_2, x_3, x_4, x_5 e x_6 que são representados pelos atributos y_1, y_2, y_3, y_4, y_5 e y_6, onde $N, M = 6$. Na Figura 2 também estão ilustrados dois biclusters hipotéticos: o primeiro formado pelos objetos x_5 e x_6 e todos os atributos $(y_1, \ldots, y_6)$, representando um bicluster com *modelo global*; e o segundo formado pelos objetos x_2, x_3, x_4 e x_5 e atributos y_4, y_5 e y_6, representando um bicluster com *modelo local*, que leva em consideração apenas um subconjunto dos atributos.

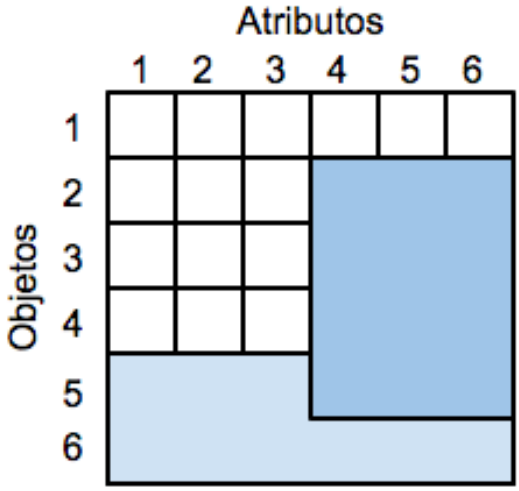


Figura 2 *– Conjunto de dados com dois biclusters encontrados.*

1.3 Hipótese

A falta de *serendipidade* nas recomendações é um problema para SRs baseados em conteúdo. Biclusterização é uma técnica capaz de expandir as possibilidades de formação de clusters, levando em consideração a análise de subconjuntos de atributos para análise de similaridade entre dados a serem considerados como pertencentes a um cluster, ou bicluster. A possibilidade de considerar a análise de subconjuntos de atributos pode levar à descoberta de objetos similares, que não são necessariamente similares, quando todo o conjunto de atributos é considerado. Sendo assim, a hipótese formulada nesse projeto é que considerar similaridades parciais entre itens a serem recomendados pode melhorar a serendipidade das recomendações e, uma forma de implementar essa estratégia em SRs,

seria por meio da aplicação de algoritmos de biclusterização sobre o conjunto de itens a serem recomendados.

1.4 Objetivos

O objetivo geral desse trabalho é amenizar a dificuldade de oferecer recomendações com *serendipidade* em SRs que se baseiam no conteúdo textual. Esse objetivo deverá ser atingido por meio da aplicação de técnicas de clusterização de texto, em particular pela aplicação de algoritmos de biclusterização.

A clusterização, nesse contexto, deve atuar como estratégia de organização do conteúdo textual, agrupando itens em biclusters. Sendo que a partir da observação dos interesses do usuário e dos biclusters à que estes itens pertencem, é possível organizar a lista recomendação com itens desses biclusters que esse usuário ainda não interagiu.

Assim, com o intuito de melhor definir o contexto de estudo neste trabalho, foram estabelecidos os seguintes objetivos específicos:

- Levantamento do referencial teórico de SRs.
  - Levantamento do estado da arte sobre o uso de aprendizado de máquina na implementação de SRs baseados em conteúdo textual.
- Levantamento do referencial teórico sobre Biclusterização e Mineração de Texto.
- Construção de um corpus de itens textuais e um conjunto de dados de recomendação.
- Implementação e teste dos algoritmos de biclusterização sobre o corpus construído.
- Modelagem do problema de recomendação baseado em conteúdo considerando a existência de biclusters de itens.
- Análise da viabilidade da modelagem proposta frente ao conjunto de dados de recomendação e frente a análise qualitativa do alcance da serendipidade.

1.5 Metodologia

A análise exploratória da literatura especilizada foi escolhida como estratégia para a aquisição de conhecimento sobre a área de SR, biclusterização e mineração de

textos. Já para o levantamento do estado da arte em SRs baseados em conteúdo textual e sobre a aplicação de técnicas de Aprendizado de Máquina, mais especificamente clustering, em SRs baseados em conteúdo textual, optou-se por aplicar a técnica de construção de revisões sistemáticas. A Revisão Sistemática (RS) é um tipo de revisão bibliográfica documentada, na qual cada passo da pesquisa é registrado seguindo critérios rigorosos, e portanto permitindo facilmente a auditoria e reprodução da pesquisa. Segundo Kitchenham (2004), a RS é um meio de avaliar, identificar e interpretar todas as pesquisas relevantes disponíveis em uma determinada área com base em questões de pesquisa. A revisão sistemas que relaciona Aprendizado de Máquina e SRs se faz útil uma vez que a aplicação de Biclusterização como estratégia de resolução do problema é o núcleo deste projeto.

A fim de permitir os testes e validações sobre a modelagem a ser proposta para verificação da hipótese formulada neste projeto, faz-se necessário a definição de um contexto para realização de uma prova de conceito. Assim, foi escolhido usar o conteúdo referente à notícias publicadas no Portal iG[1]. Trata-se de um portal de notícias brasileiro muito conhecido, com um volume de notícias bastante grande e com alguma estrutura de classificação de notícias já existente. Essas características conferem liberdade para a configuração de experimentos de diferentes naturezas, como experimentos considerando determinadas classes de notícias, tipos de notícias ou datas de publicação das notícias.

A partir do conteúdo de notícias do Portal iG deve ser construído um corpus de dados textuais, categorizados de acordo com as categorias já usadas no referido portal. Todo o conteúdo do corpus deve passar por rotinas de pré-processamento comuns na área de Mineração de Texto: *tokenização*, filtragem de *stopwords*, remoção de sufixos (*stemming*), representação da relação "termos × documentos" usando estratégias de frequência termos (TF-IDF) e *n-grams*.

Também, o Portal iG possui informações históricas e anônimas referentes ao registro de navegação de usuários. Esse registro permite a construção de um conjunto de dados de preferências, que pode ser usado para realização de *testes offline* entre recomendações oferecidas pela abordagem proposta e a navegação real realizada por um conjunto de usuários, utilizando métricas como precisão e revocação (JANNACH et al., 2011).

Os resultados da aplicação dos algoritmos de análise de texto via biclusterização deverão ser validados fazendo a técnicas de avaliação internas, para a verificação da consistência dos biclusters encontrados (SANTAMARíA; MIGUEL; THERóN, 2007), e externas

[1] http://ig.com.br/

(HOCHREITER et al., 2010), avaliando o quanto os biclusters encontrados estão em consenso com as classes de notícias (HOCHREITER et al., 2010).

Em tempo, estuda-se a possibilidade de aplicar técnicas de *ensemble* com algoritmos clássicos de clusterização a fim de substituir o uso de algoritmos de biclusterização na análise da correlação parcial dos atributos descritivos dos itens sob recomendação. Ainda, modelos escondidos de Markov podem ser aplicados na modelagem da recomendação sobre o conjunto de dados de recomendação a fim de gerar instâncias de comparação para avaliação da análise proposta neste projeto.

1.6 Organização do documento

Este documento é composto por quatro capítulos incluindo esta introdução. No Capítulo 2 conceitos fundamentais necessários para a compreensão deste projeto são apresentados, incluindo conceitos sobre SRs, biclusterização e mineração de texto; o Capítulo 3 apresenta uma revisão sistemática sobre aprendizado de máquina aplicado à SRs baseados em conteúdo; por fim, o capítulo 4 apresenta em detalhes a proposta desse projeto de mestrado, um relato sobre os resultados já obtidos e o cronograma que guia a realização das próximas atividades.

Capítulo 2

Conceitos Fundamentais

Este capítulo introduz os conceitos fundamentais para o entendimento da proposta de trabalho de mestrado apresentada neste documento. Conceitos nas áreas de Sistemas de Recomendação (Seção 2.1), Biclusterização (Seção 2.2) e Mineração de Texto (Seção 2.3) são apresentados.

2.1 Sistemas de Recomendação

A grande maioria das pessoas que usam a web muito provavelmente já interagiu com algum Sistema de Recomendação (SR), e por isso, o conceito que define esse tipo de sistema é intuitivo. Porém, por ser uma área forma de estudo relativamente nova (LOPS; GEMMIS; SEMERARO, 2011), ainda diferentes autores usam diferentes definições, visto que a interpretação do que consiste um SR difere dependendo do ponto de vista do autor (LEINO, 2014).

Resnick e Varian (1997), autores que criaram o termo Sistemas de Recomendação (NEUMANN, 2007; LEINO, 2014), argumentam que SRs servem para ajudar em processos de tomada de decisão do dia-a-dia, como quais itens comprar, quais músicas ouvir, ou quais notícias ler. Além disso, Resnick e Varian (1997) provê uma taxonomia na área de Sistemas de Recomendação:

- Conteúdo recomendado: os itens que são recomendados pelo Sistema de Recomendação. Por exemplo: produtos, músicas, notícias e/ou etc.
- Entrada dos usuários: as interações que os usuários realizam com os itens são a entrada para um SR, e elas podem ser implícitas (o usuário u leu a notícia I) ou explícitas (o usuário u classificou o filme I como 5 estrelas).
- Alvo da recomendação: os itens podem ser recomendados especificamente para um usuário (personalizado), direcionados para um grupo de usuários ou a todos os usuários (não-personalizado).

- Técnicas para recomendação (agregações): as estratégias e os algoritmos que os SRs usam para criar as recomendações.

- Uso das recomendações: trata-se de como mostrar as recomendações para os usuários. Por exemplo: por meio do filtro de recomendações negativas, ordenando por um fator de classificação 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 ele, dentro de uma grande quantidade de opções. Isso faz com que SRs que provêm recomendações não-personalizadas deixem de se adequarem à definição de SR.

Formalmente, Burke (2002, 2007) define SRs como um conjunto de itens $\mathcal{I}$ que podem ser recomendados, um conjunto de usuários U cujas preferências são conhecidas, um usuário u pra o qual as recomendações são geradas, e algum item i para o qual se quer predizer a preferência para u. Adomavicius e Tuzhilin (2005) extendem a definição de SR com uma função de utilidade f que mede o quão útil é o item i para o usuário u: $f : \mathcal{I} \times U \rightarrow R$, em que $R = \{r_{u_1,I_1}, \ldots, r_{u_1,I_m}, \ldots, r_{u_1,I_M}, \ldots, r_{u_n,I_1}, \ldots, r_{u_n,I_M}, \ldots, ru_N, I_1, \ldots, r_{u_N,I_m}, \ldots, r_{u_N,I_M}\}$ é um conjunto ordenado com valores faltantes, sendo r_{u_n,i_m} 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 à um usuário u (LOPS; GEMMIS; SEMERARO, 2011).

Em síntese, um Sistema de Recomendação tem a função de auxiliar os usuários de uma aplicação a interagir com itens, provendo sugestões de com 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, e do uso delas surgem os tipos de SRs. Os tipos de SRs diferem quanto ao domínio, informações usadas para recomendação, e principalmente nas propriedades em que cada tipo se destaca.

Jannach et al. (2011) provêm uma taxonomia que categoriza os SRs em quatro diferentes tipos:

- *Filtragem colaborativa*, o primeiro tipo de SR que foi implementado (RESNICK; VA-

RIAN, 1997), tem como idéia básica encontrar outros usuários $u_{1,...,Z}$ em U, sendo $Z < N$ e não necessariamente em ordem, com preferências semelhantes à u, e então recomendar itens com os quais $u_{1,...,Z}$ interagiram e que u ainda não interagiu, estabelecendo alguma métrica para estimar f. Medidas de similaridade geralmente usadas nesse contexto incluem *Correlação de Pearson* e *Similaridade dos Cossenos*. Também são usadas 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 e itens em $\mathcal{I}$, 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 $\mathcal{I}$ 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 $\mathcal{I}$. São estabelecidas medidas de similaridade para estimar o quanto as necessidades do usuário são atendidas pelas recomendações (JANNACH et al., 2011; LOPS; GEMMIS; SEMERARO, 2011; BURKE, 2007).

- *Híbrido*, é capaz de combinar as vantagens de cada tipo de SR para suprir as limitações associadas a cada um deles. A dificuldade, neste caso, está em como combinar as diferentes técnicas para recomendação (JANNACH et al., 2011; BURKE, 2007). Burke (2007) identificou sete tipos de SRs Híbridos em uma revisão da literatura: *Weighting*, atribui um peso para cada algoritmo a fim de ponderar as recomendações; *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. Em Jannach et al. (2011), Adomavicius e Tuzhilin (2005), Burke (2002), Lops, Gemmis e Semeraro (2011) os problemas comuns encontrados no desenvolvimento de SRs são nomeados: *novo usuário* (*user cold-start*), *novo item* (*item cold-start*), *esparsidade*, *sobre-especialização* e *análise de conteúdo limitada*.

Os problemas de *novo usuário* e *novo item* são similares. Basicamente, enquanto o primeiro trata da dificuldade de gerar recomendações para novos usuários, o segundo 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 tenha 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 e itens, um item novo, que não teve nenhuma 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 combinam o perfil de preferências de um usuário com os itens em $\mathcal{I}$, 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 ($U \times \mathcal{I}$) 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, a qual precisa ser suficiente para discriminá-los (LOPS; GEMMIS; SEMERARO, 2011). Em SRs baseados em conteúdo é comum haver a necessidade de limitar a representação de itens, tanto pelo número de atributos quanto pelo conteúdo que essas podem representar. Por exemplo, no contexto de notícias na web, se a representação for um vetor com o número de ocorrências de cada palavra, perde-se 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 tipos de SRs foram discutidos juntamente com uma variedade de problemas relacionados a eles. Nesse contexto, faz-se a importância da discussão de como avaliar se uma estratégia adotada no desenvolvimento de um SR 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 aleatoriamente, 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 mais simples, esses tipos de experimentos podem ser usados, por exemplo, para selecionar os melhores algoritmos a serem aplicados no desenvolvimento de um SR. 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 àquelas adotadas em Aprendizado de Máquina, como avaliação da *matriz de confusão* e de medidas derivadas como *precisão*, *revocação*, *f1-score*, *cross-validation* e etc.

Na realidade, a estratégia de avaliação via *matriz de confusão* pode 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, a predição é considerada errada (falso positivo); contrariamente, se o SR não fizer essas sugestões, é considerada uma emissão correta (verdadeiro negativo); por fim, se o SR não sugerir itens que o usuário tem preferência, é considerada 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)[1]. A definição 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]Essa métrica foi a utilizada na competição Netflix Prize (*http://www.netflixprize.com/*), que teve grande repercursão na academia e na indústria

2.2 Biclusterização

Técnicas e algoritmos de biclusterização são usadas, principalmente, no contexto de expressão genética. No entanto, algoritmos de biclusterização se fazem úteis 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): seja 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 é 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).

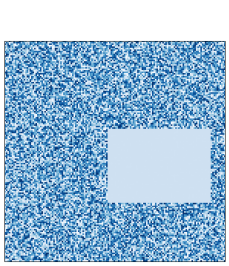
- *Biclusters com evoluções coerentes*, têm seus valores com evoluções coerentes, por exemplo, um bicluster com $a_{i4} \leq a_{i3} \leq a_{i2} \leq a_{i1}$ tem valores com evolução coerente na coluna (MADEIRA; OLIVEIRA, 2004). Seus valores podem ser gerados por uma função geradora de valores com evolução 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 diferem quanto as suas estruturas. Cada algoritmo usado para implementar biclusterização faz uma suposição da estrutura de biclusters que é capaz de encontrar. A Figura 3 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.

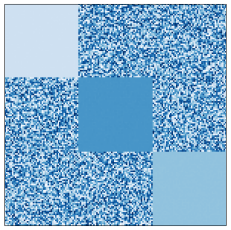
2.2.2 Algoritmos para biclusterização

Diversos algoritmos para encontrar biclusters, de diferentes tipos e estruturas, foram propostos na literatura (TANAY; SHARAN; SHAMIR, 2005; MADEIRA; OLIVEIRA, 2004).

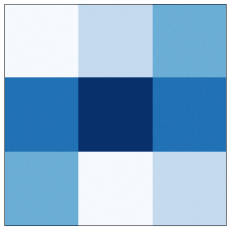
Um dos algoritmos de biclusterização mais comum e simples. que encontra biclusters com valores coerentes, em estrutura com sobreposição e arbitrariamente posicionados, é o *Coupled Two-way Clustering* (CTWC) (GETZ; LEVINE; DOMANY, 2000). O algoritmo CTWC é capaz de encontrar biclusters através da clusterização de objetos e atributos (linhas e colunas), separadamente. O algoritmo de clusterização usado por Getz, Levine e Domany (2000) foi o *Superparamagnetic Clustering* (SPC), o qual é capaz de determinar o número de clusters automaticamente, e com uma estratégia de clusterização hierárquica *top-down* é capaz de gerar clusters estáveis (GETZ; LEVINE; DOMANY, 2000). O SPC tem como entrada uma matriz de similaridade e um parâmetro temperatura, que controla o quão estáveis serão os clusters que o algoritmo gerará. Assim, o CTWC encontra clusters estáveis de linhas e colunas através do SPC, e iterativamente executa o SPC nos clusters de linhas e colunas encontrados, mantendo na memória um par do subconjunto de linhas e do subconjunto de colunas (biclusters), assim como os clusters estáveis de linhas e colunas, separadamente.



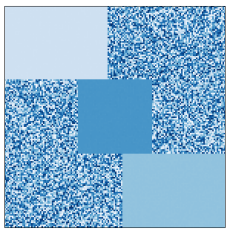
(a) *Bicluster único*



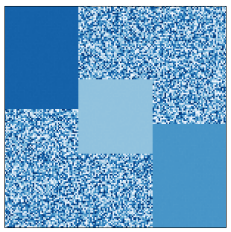
(b) *Biclusters com linhas e colunas exclusivas*



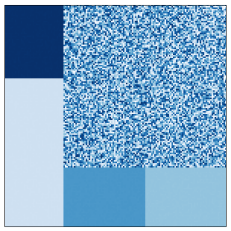
(c) *Biclusters com estrutura de tabuleiro de xadrez*



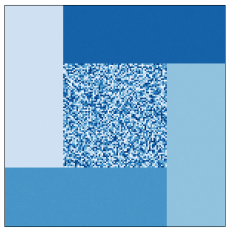
(d) *Biclusters com linhas exclusivas*



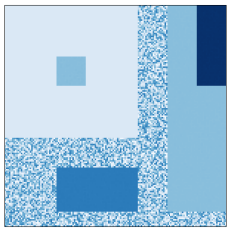
(e) *Biclusters com colunas exclusivas*



(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 3 *– Diferentes estruturas de biclusters, quadrados com cores sólidas representam biclusters*

Já o algoritmo de Cheng e Church (2000) é capaz de encontrar o mesmo tipo de bicluster que o algoritmo CTWC, porém usando uma estratégia gulosa: biclusters com valores coerentes e estrutura com sobreposição e arbitrariamente posicionados. Este algoritmo esta sendo objeto de estudo desse projeto de mestrado para aplicação em dados textuais e por isso segue aqui descrito em mais detalhes. Nesse algoritmos, para encontrar biclusters, ou δ-biclusters, na matriz A, os autores definem o *Resíduo Quadrático Médio* (RQM):

$$\begin{aligned} H_{IJ} &= \frac{1}{|I||J|} \sum_{i,j \in I,J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2 \\ H_{iJ} &= \frac{1}{|J|} \sum_{j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2 \\ H_{Ij} &= \frac{1}{|I|} \sum_{i \in I} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2 \end{aligned} \tag{2.3}$$

em que

$$a_{iJ} = \frac{1}{|J|}\sum_{j\in J} a_{ij}, \quad a_{Ij} = \frac{1}{|I|}\sum_{i\in I} a_{ij}, \quad a_{IJ} = \frac{1}{|I||J|}\sum_{i,j\in I,J} a_{ij} \tag{2.4}$$

onde H_{IJ} é o RQM de uma submatriz A_{IJ}, H_{iJ} o RQM da linha i, H_{Ij} o RQM da coluna j, a_{iJ} a média dos valores da linha i, a_{Ij} a média dos valores da coluna j e a_{IJ} a média dos valores da submatriz A_{IJ}, definida pelos subconjuntos I e J.

Então, um bicluster perfeito A_{IJ} teria o RQM $H_{IJ} = 0$, pois $a_{ij} = a_{ij}, \forall i, j \in I, J$, fazendo $a_{iJ} = a_{Ij} = a_{IJ}$. No entanto, se apenas minizar o RQM, um bicluster com apenas um elemento seria perfeito, o que pode não refletir a realidade. Além disso, em conjunto de dados reais existe ruído, podendo esconder o bicluster perfeito.

Para encontrar biclusters, ou δ-biclusters, Cheng e Church (2000) usam uma estratégia gulosa que retira linhas e colunas, visando a minimização do RQM, respeitando um parâmetro δ, que é calibrado pelo usuário. Então, um bicluster é encontrado quando o RQM de uma submatriz A_{IJ} é $H_{IJ} \leq \delta$, para algum $\delta \geq 0$. As etapas de remoções de elementos da matriz são apresentadas nos algoritmos 1 e 2.

Input: A, I, J, δ
Output: I, J onde $H_{IJ} \leq \delta$
while $H_{IJ} > \delta$ **do**
    encontre a linha $l_{max} = \arg\max_{i\in I} H_{iJ}$;
    encontre a coluna $c_{max} = \arg\max_{j\in J} H_{Ij}$;
    if $l_{max} > c_{max}$ **then**
        remova l_{max} do subconjunto I;
    else
        remova c_{max} do subconjunto J;

Algoritmo 1: Remove uma linha ou coluna a cada iteração.

Input: A, I, J, δ, α
Output: I, J onde $H_{IJ} \leq \delta$
while $H_{IJ} > \delta$ **do**
    remova $i \in I$ onde $H_{iJ} > \alpha \cdot H_{IJ}$;
    remova $j \in J$ onde $H_{Ij} > \alpha \cdot H_{IJ}$;
    encontre a coluna $c_{max} = \arg\max_{j\in J} H_{Ij}$;
    if *não houve nenhuma remoção* **then**
        chame o algoritmo 1

Algoritmo 2: Remove múltiplas linhas e colunas de A a cada iteração.

O algoritmo 2 é usado para acelerar o processo de busca de um δ-bicluster, convergindo mais rapidamente para uma solução quanto maior for o parâmetro α, em que $\alpha \geq 0$. Ainda, para amenização do problema de encontrar δ-biclusters perfeitos com apenas um elemento, ou poucos elemento, é utilizado o algoritmo **??**, que adiciona nós sem aumentar o RQM do bicluster.

Input: A, I, J
Output: I', J' onde $H_{I'J'} \leq H_{IJ}$
while $H_{IJ} > \delta$ **do**
  compute a_{iJ} e a_{Ij} para todo i, j e H_{IJ};
  remova $j \notin J$ onde $\frac{1}{|I|}\sum_{j \in J}(a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2 \leq H_{IJ}$;
  recompute a_{iJ} e H_{IJ};
  remova $i \notin I$ onde $\frac{1}{|J|}\sum_{i \in I}(a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2 \leq H_{IJ}$;
  if *não houve nenhuma adição* **then**
    $I' \longleftarrow I$;
    $J' \longleftarrow J$;

Algoritmo 3: Adiciona linhas e colunas de A_{IJ} a cada iteração.

Por fim, o algoritmo **??** é a consolidação dos algoritmos **??**, 2 e 1 e a iteração para encontrar k δ-biclusters, um a um, sendo k fornecido pelo usuário.

Input: A, k, δ, α
Output: k δ-biclusters
$A' \leftarrow A$;
for 1 **to** k **do**
  $B \leftarrow$ algoritmo 2 com A', δ, α;
  $C \leftarrow$ algoritmo 1 com B, δ;
  $D \leftarrow$ algoritmo **??** com A, C;
  reporte D como uma solução;
  adicione ruído em A' para a submatriz D;

Algoritmo 4: Algoritmo Cheng & Church, encontra k δ-biclusters.

Além dos algoritmos apresentados, existem outros algoritmos que são capazes de encontrar outros tipos de biclusters (seção 2.2.1), além de serem recentes (FRANçA; ZUBEN, 2010; YANG; LESKOVEC, 2013; HOCHREITER et al., 2010; CABANES; BENNANI; FRESNEAU, 2012), mostrando que ainda há interesse na área de pesquisa de biclusterização.

2.2.3 Avaliação de biclusterização

Para determinar parâmetros, descobrir a qualidade e/ou estabilidade dos biclusters encontrados por algoritmos, é necessário estabelecer métricas de avaliação. Existem

duas maneiras de avaliar biclusters (HOCHREITER et al., 2010): *interna*, usa os dados dos resultados dos algoritmos, juntamente com métricas de qualidade e/ou estabilidade, para avaliar as soluções geradas; *externa*, utiliza os dados reais das soluções de biclusters de um conjunto de dados, usando estratégias para comparação, obtendo assim, maior confiança nas soluções.

A avaliação interna pode não ser tão precisa quanto a avaliação externa, porém é útil para descobrir parâmetros ótimos. Apesar de PreliĆ et al. (2006) sugerirem não usar avaliações internas, por não estar claro como extender noções de separação e homogeinidade, Santamaría, Miguel e Therón (2007) descreveu métricas de consistência para verificando se um bicluster é consistente com a sua definição, seja aditiva, multiplicativa e/ou constante, fazendo uma comparação dos elementos do bicluster:

$$\begin{aligned} C_l(A_{IJ}) &= \frac{1}{|I|}\sum_{i=1}^{|I|-1}\sum_{j=i+1}^{|I|}\sqrt{\sum_{k=1}^{|J|}(a_{ik}-a_{jk})^2} \\ C_c(A_{IJ}) &= \frac{1}{|J|}\sum_{i=1}^{|J|-1}\sum_{j=i+1}^{|J|}\sqrt{\sum_{k=1}^{|I|}(a_{ki}-a_{kj})^2} \end{aligned} \tag{2.5}$$

em que $C_l(A_{IJ})$ é o índice de consistência das linhas do bicluster A_{IJ} e $C_c(A_{IJ})$ é o índice de consistência das colunas do bicluster A_{IJ}. Ainda, a consistência do bicluster inteiro C pode ser definida pela média:

$$C(A_{IJ}) = \frac{|I|\cdot C_l + |J|\cdot C_c}{|I|+|J|} \tag{2.6}$$

Uma das métricas externas que são usadas para comparar biclusters encontrados com biclusters reais em um conjunto de dados, é a métrica *concensus score* (HOCHREITER et al., 2010). Essa métrica calcula a maximização das similaridades entre biclusters encontrados e reais, usando o *índice de Jaccard* como medida de similaridade e o algoritmo Húngaro para solucionar o problema de maximização. A saída da avaliação é um *score* $\in [0,1]$, em que 0 significa que os biclusters comparados são totalmente diferentes, e 1 o inverso.

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 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 e 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*, que incluem algoritmos para a descoberta de padrões, tendências e conhecimentos por meio de técnicas e algoritmos; *componentes de apresentação*, que incluem interfaces para o usuário, apresentando visualizações dos conhecimentos gerados na etapa anterior; e *técnicas de refinamento*, também descritas como uma fase de pós-processamento, que incluem 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 termos, ou *Vector Space Model* (SALTON; WONG; YANG, 1975), é a representação clássica usada para representar documentos textuais (SEBASTIANI, 2002; LOPS; GEMMIS; SEMERARO, 2011). Cada dimensão desse vetor está associada a um termo, sendo que todas as dimensões representam todos os termos do conjunto de documentos. Formalmente, há 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 termos $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$, em que t_j representa um termo e m o número de termos presentes em todos os documentos. Representando a frequência de um termo pelo número de vezes que t_j aparece em um documento d_i, denotado por $ft(t_j, d_i)$, o vetor de termos pode ser construído e representado da seguinte forma: $\vec{vt}_{d_i} = (TF(t_1, d_i), TF(t_2, d_i), \ldots, TF(t_m, d_i))$. Salton, Wong e Yang (1975) argumentam que a representação textual de documentos em

vetor de termos é suficiente para separar documentos. Ao invés de frequência de termos, também é usado, a representação binária (SEBASTIANI, 2002), ou seja, t_j aparecendo em d_i corresponde à entrada 1 na dimensão j em $\vec{vt}_{d_i}$. Há também outros métodos para representação textual, como *n-gramas* e *ontologias* (LOPS; GEMMIS; SEMERARO, 2011).

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

$$\begin{aligned} \textit{TF-IDF}(t_j, d_i) &= TF(t_j, d_i) \cdot IDF(t_j) \\ \textit{TF-IDF}(t_j, d_i) &= TF(t_j, d_i) \cdot \left(log_2 \frac{n}{DF(t_j) + 1} \right) \end{aligned} \tag{2.7}$$

em que $IDF(t_j)$ representa a frequência de documentos inversa do termo t_j, e $DF(t_j)$ a frequência de documentos que contém t_j. Essa normalização faz com que a frequência dos termos que aparecem em muitos documentos seja reduzida, e a frequência dos termos 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 termos, o primeiro processo a ser realizado é a *tokenização*, que cria um dicionário de termos 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 termos do conjunto $\mathcal{T}$ 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 determinados termos de $\mathcal{T}$ a partir de uma lista, é chamada de *stopwords*. Também são usadas outras técnicas, como a eliminação de termos com a frequência muito alta ou muito baixa.

2.3.1.4 Stemming

A fim de reduzir a ambiguidade de termos, o método de *stemming* é capaz de juntar, em uma única forma, termos relacionados (MINER et al., 2012). Por exemplo, o verbo *fazer* pode se apresentar em diversas formas, como *fazendo*, *fez*, etc. Esse processo é capaz de aumentar a capacidade da representação em distinguir ou identificar documentos, além de reduzir a dimensionalidade, reduzindo também a esparsidade.

2.3.1.5 Redução de Dimensionalidade

A representação em vetor de termos pode resultar em vetores esparsos num espaço de alta dimensão, que pode fazer com que algoritmos sofram do problema de *Maldição de Dimensionalidade*, que diz respeito à perda de densidade em espaços de alta dimensão, isto significa que medidas de distância se tornam incapazes de detectar padrões em um conjunto de dados (HAYKIN, 2008). 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 termos, diminuindo a dimensionalidade e esparsidade, superando o problema de *Maldição de Dimensionalidade*.

Capítulo 3

Sistemas de Recomendação baseados em Conteúdo e Aprendizado de Máquina

Os Sistemas de Recomendação baseados em Conteúdo (SRbC) têm fortes relações com a área de Recuperação de Informação (ADOMAVICIUS; TUZHILIN, 2005; JANNACH et al., 2011) e Aprendizado de Máquina (ADOMAVICIUS; TUZHILIN, 2005; LOPS; GEMMIS; SEMERARO, 2011), para representação de itens e perfis de usuários, e aprendizado do perfil do usuário. Basicamente, este tipo de sistema analisa o conteúdo de diversos itens, extraindo atributos para representação, com esses mesmos atributos (ou às vezes até mais (CAPELLE et al., 2012)) representa-se o perfil do usuário. Sabendo os interesses dos usuários, através do perfil construído, o sistema seleciona itens que o usuário ainda não consumiu e que sejam relacionados com os seus interesses.

3.1 Arquitetura de Sistemas de Recomendação baseados em Conteúdo

Lops, Gemmis e Semeraro (2011) propõem uma arquitetura para o desenvolvimento de SRbC, a qual separa o processo de recomendação em três fases (Figura 4). O analisador de conteúdo tem como entrada os itens não estruturados, assim, através de técnicas de Mineração de Dados, os itens são representados de forma estruturada. Uma representão comum, no contexto de conteúdo textual, é o *Vector Space Model* (ADOMAVICIUS; TUZHILIN, 2005; LOPS; GEMMIS; SEMERARO, 2011; JANNACH et al., 2011) (Seção 2.3.1). Onde representa-se o vetor de termos do documento d_i por $\vec{vt}_{d_i} = (ft(t_1, d_i), ft(t_2, d_i), \ldots, ft(t_m, d_i))$, em que t_i é um termo, para então usar a representação de TF-IDF.

Com a representação dos itens estruturada realizada, ocorre a representação dos perfis, que geralmente é baseado na representação dos itens, ou seja, o perfil do usuário u é

dado por $\{(d_1, r_{u,d_1}), \ldots, (d_j, r_{u,d_n})\}$, sendo r_{u,d_n} o quão o usuário u gostou do documento d_n, seja pela manifestação explícita, por exemplo em que o usuário avaliou o documento, ou pela manifestação implícita, tendo como exemplo quando o usuário lê uma notícia, fica muito tempo na página, etc. Finalmente, é possível aprender os perfis dos usuários utilizando de técnicas e algoritmos de Aprendizado de Máquina (ADOMAVICIUS; TUZHILIN, 2005; LOPS; GEMMIS; SEMERARO, 2011; JANNACH et al., 2011), por exemplo, para prever se o usuário gosta ou não de um determinado item (classificação).

A terceira fase é o Componente de Filtragem, que basicamente recebe a saída do classificador, seleciona os itens mais relevantes para os usuários, e apresenta uma lista de recomendações. Geralmente, essa lista é ordenada e apresentada os top N itens mais relevantes.

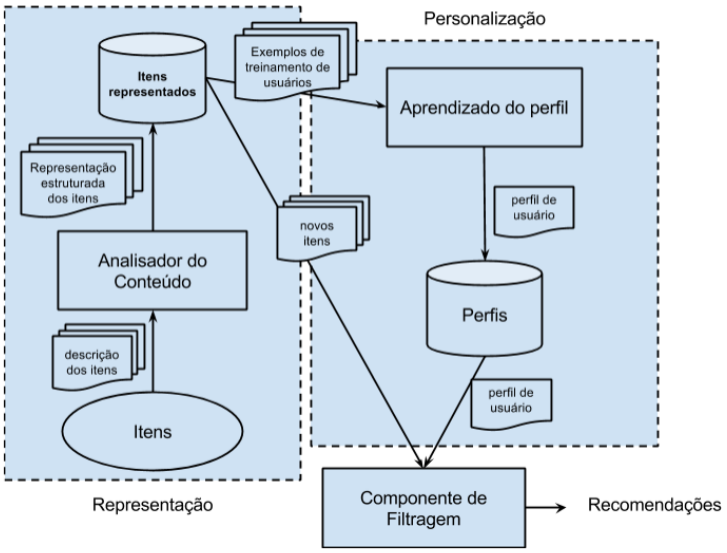


Figura 4 *– Arquitetura de um CBRS*

Essa arquitetura apresentada foi adaptada de (LOPS; GEMMIS; SEMERARO, 2011), para maior compreendimento e maior adequação com esse trabalho. As subseções 3.1.1, 3.1.2 e 3.1.3 fazem uma revisão da literatura das técnicas de Aprendizado de Máquina aplicadas aos módulos de um SRbC.

3.1.1 Analisador de Contexto

O Analisador de Contexto tem como função representar o item de uma maneira estruturada, a Figura 5 explica como é construída essa etapa. A entrada são os itens, que são pré-processados, para então aplicar técnicas de Aprendizado de Máquina (Redução de Dimensionalidade ou Tarefas de Aprendizado de Máquina). Esta etapa compreende o

foco deste trabalho.

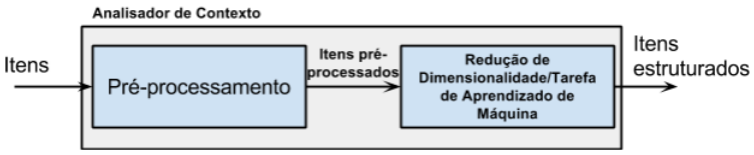


Figura 5 *– Analisador de Contexto.*

Na etapa de pré-processamento, é comum o uso da representação TF-IDF, sendo que existem estudos que propõem outros métodos de representação (CAPELLE et al., 2012; MOERLAND et al., 2013): SF-IDF (*Semantics Frequency-Inverse Document Frequency*) e SF-IDF+, que obtiveram melhores resultados nos testes apresentados, além disso, Cleger-Tamayo, Fernández-Luna e Huete (2012) considerou o uso do TF-IDF, mas não resultou em melhores performances. No entanto, em um SR no contexto de artigos científicos (BEEL et al., 2013), foram realizados diversos testes considerando diversos parâmetros e configurações diferentes de representação, resultados mostraram que a configuração com TF-IDF performou melhor que outras. Outras técnicas de Mineração de Dados também são utilizadas, como *stopwords*, lematização e descarte de termos com frequência abaixo de um limiar.

Na etapa de Redução de Dimensionalidade/Tarefa de Aprendizado de Máquina faz-se o uso extensivo dos algoritmos: LSA (*Latent Semantic Analysis*) (TARAGHI et al., 2013; DOMINGUES et al., 2012; SPAETH; DESMARAIS, 2013), LSI (*Latent Semantic Indexing*) (SAAYA et al., 2013), LDA (*Latent Dirichlet Allocation*) (TANTANASIRIWONG, 2012; QU; LIU, 2012; WANG et al., 2012; VAZ; Martins de Matos; MARTINS, 2012), que pode ser visto como uma extensão de biclusterização (SKILLICORN, 2012), e Variáveis Latentes (CLEGER-TAMAYO; FERNáNDEZ-LUNA; HUETE, 2012), que podem ser vistos por resolver uma tarefa de agrupamento, que é agrupar termos em grupos que são chamados de tópicos (WANG et al., 2012). O desafio é escolher o número de tópicos, por isso, grande parte dos trabalhos que fazem o uso dessas técnicas, realizam testes variando o número de tópicos. Esses testes geralmente mostram que esses algoritmos ajudam na performance da recomendação (CLEGER-TAMAYO; FERNáNDEZ-LUNA; HUETE, 2012; TANTANASIRIWONG, 2012; SAAYA et al., 2013; SPAETH; DESMARAIS, 2013; VAZ; Martins de Matos; MARTINS, 2012).

Na representação estruturada dos itens, alguns estudos propõem representações no contexto de notícias e artigos científicos (BIELIKOVA; KOMPAN; ZELENIK, 2012; LOPS et al., 2013): são diferenciadas as relevâncias de cada um dos atributos textuais (exemplo

título, conteúdo, categoria e etc.), por meio de pesagem. Estudos mostraram que com os pesos apropriados é possível melhorar a qualidade da recomendação. Atributos que não são textuais também são usados no contexto de notícias, como no SR de notícias para celular em (YEUNG; YANG; NDZI, 2012), que o tempo da notícia modifica o vetor que representa o item, fazendo a multiplicação por um fator α. o SR de livros (VAZ; Martins de Matos; MARTINS, 2012) que tem como objetivo representar o estilo de escrita de cada autor, faz o uso de atributos como: o tamanho do documento, n-gramas e *vocabulary richness*.

3.1.2 Aprendizado do Perfil

Nesta etapa é onde o perfil do usuário é representado e aprendido pelo SR. Na maioria das vezes o perfil é representado por um vetor de documentos de tamanho k, que o usuário visitou ou apresentou um feedback positivo $d_{prefs} = \{d_{pref_1}, d_{pref_2}, \ldots, d_{pref_k}\}$. Há outras formas de representação, como em Yeung, Yang e Ndzi (2012), que além da anterior, incorpora informações demográficas, tratando o problema de *user cold-start* em SRsbC. Vaz, Martins de Matos e Martins (2012) propõem uma representação para o perfil do usuário usando um método da área de Recuperação de Informação: algoritmo de *Rocchio*, onde cada documento d_{pref_i} é classificado pelo usuário em positivo ou negativo (como exemplo, gostou ou não gostou), assim o algoritmo faz uma mistura dos exemplos positivos e negativos, com um peso diferente para cada tipo de exemplo, obtendo um vetor. Então, esse vetor é comparado com vetores de itens (usando similaridade dos cossenos), para obter itens semelhantes. Variando os parâmetros, os autores chegaram na conclusão que, incorporando exemplos negativos na representação do perfil para treinamento do algoritmo, piora a qualidade das recomendações.

O aprendizado do perfil do usuário é como um problema de classificação, onde o vetor de exemplos é dado por $X = \{d_{pref_i}, y_i^{+-}\}$, sendo que y_i^{+-} representa o rótulo, ou seja, se d_{pref_i} é um item que o usuário gostou$(+)$ ou não$(-)$. Então, é treinado um classificador que irá classificar itens que o usuário ainda não consumiu, para saber se é um item que o usuário irá consumir/gostar. Diversos algoritmos são usados para resolver esse tipo de problema: Redes Bayesianas (YEUNG; YANG; NDZI, 2012; CLEGER-TAMAYO; FERNáNDEZ-LUNA; HUETE, 2012), Naïve Bayes (LEE et al., 2012; SEMERARO et al., 2012), SVM (*Support Vector Machine*) (TANTANASIRIWONG, 2012; LEE et al., 2012).

Existem trabalhos que tratam o aprendizado do perfil com técnicas de Aprendizado Semi-Supervisionado, Lee et al. (2012) faz uso de comitê de máquinas para construir

um modelo de Aprendizado de Máquina que classifica apenas classes positivas, visando classificar se o usuário de um e-commerce irá gostar ou não de um produto. Primeiramente, classifica exemplos sem rótulo, para depois entrarem no modelo final que agrega todos os exemplos (SVM ou Naïve Bayes). Foi verificado que o SR proposto trata o problema de poucos dados para a identificação do perfil do usuário, pois não necessita apenas de dados rotulados.

É possível tratar o problema de aprendizado do perfil como um problema de clusterização (DAVOODI; KIANMEHR; AFSHARCHI, 2012; BIELIKOVA; KOMPAN; ZELENIK, 2012). Davoodi, Kianmehr e Afsharchi (2012) apresentam um SR de especialistas, que representa o perfil dos usuários com semântica e constrói uma *Rede Social*, para então, usar o algoritmo de clusterização (*k-means*) para encontrar perfis de usuário. Além desse, (BIELIKOVA; KOMPAN; ZELENIK, 2012) apresenta um SR de notícias que faz o uso de clusterização hierárquica, tendo como medidade de similaridade a similaridade dos cossenos e índice de jaccard. Com uma abordagem *bottom-up* de agrupamento e uma estrutura de árvore binária, é realizado a clusterização das notícias: as folhas representam as notícias e os nós pais, clusters que representam temas das notícias. O usuário desse sistema é representado por caminhos nesta árvore construída, podendo ser recomendados diversas notícias dentro de diversos tópicos, que podem surpreender o usuário, amenizando o problema de serendipidade em SRsbC.

3.1.3 Componente de Filtragem

Essa é a etapa mais simples, por ser na maioria das vezes, apenas uma filtragem das recomendações já calculadas na etapa de Aprendizagem do Perfil, essa estratégia é apresentada em Cleger-Tamayo, Fernández-Luna e Huete (2012), Qu e Liu (2012), Wang et al. (2012), Davoodi, Kianmehr e Afsharchi (2012), Mannens et al. (2011), Semeraro et al. (2012). Outra estratégia simples é a determinação de um limiar (CAPELLE et al., 2012; LOPS et al., 2013), ou seja, os valores da lista de recomendação gerada são filtradas pelo limiar estabelecido. Nos estudos apresentados, foram encontrados muitos SRs Híbridos, que faziam uma outra abordagem para a filtragem, usando uma combinação dos métodos de Filtro Colaborativo e baseados em conteúdo (LOPS et al., 2013; QU; LIU, 2012; DOMINGUES et al., 2012; SPAETH; DESMARAIS, 2013; VAZ; Martins de Matos; MARTINS, 2012).

O trabalho desenvolvido por Bielikova, Kompan e Zelenik (2012) foi o único que apresentou uma estratégia diferente para o Componente de Filtragem, com a árvore

binária montada, todas as notícias dos menores para os maiores grupos, que não foram lidas pelo usuário, foram separadas para a recomendação. Então, é construída uma matriz com as recomendações, sendo as linhas ordenadas pelos grupos menores para os grupos maiores, e as colunas ordenadas pelas notícias mais recentes. Assim, cada coluna é transformada em um vetor, concatenando-os e formando uma lista que é apresentada para o usuário.

Capítulo 4

Proposta

A proposta desse projeto de mestrado envolve a aplicação de algoritmos de biclusterização para o problema de recomendação baseada em conteúdo textual, a qual, segundo a revisão sistemática que esta sendo realizada, foi muito pouco explorada: apenas um entre 1 943 estudos primários relevantes nas bases de dados (não publicado) se refere à aplicação de biclusterização em conteúdo textual.

Com maior especificidade, a proposta desse projeto (Figura 6) é definida pela utilização de técnicas de mineração de texto para o processamento das notícias presentes no corpus iG (Subseção 4.1.1), após a estruturação do texto, serão realizadas diversas representações (TF-IDF, TF-IDF normalizado e n-gramas) para a criação de biclusters com algoritmos de biclusterização, realizando experimentos para verificar a consistência dos biclusters criados pelos diferentes algoritmos aplicados. Assim, será definida uma estratégia para gerar uma lista de recomendações e comparar com a base de cliques iG (Subseção 4.1.2), utilizando métricas de SRs para gerar experimentos e avaliar as recomendações geradas. Além disso, será feita uma comparação com um SR do tipo filtro colaborativo, para comparar a *serendipidade* entre o método proposto e o filtro colaborativo.

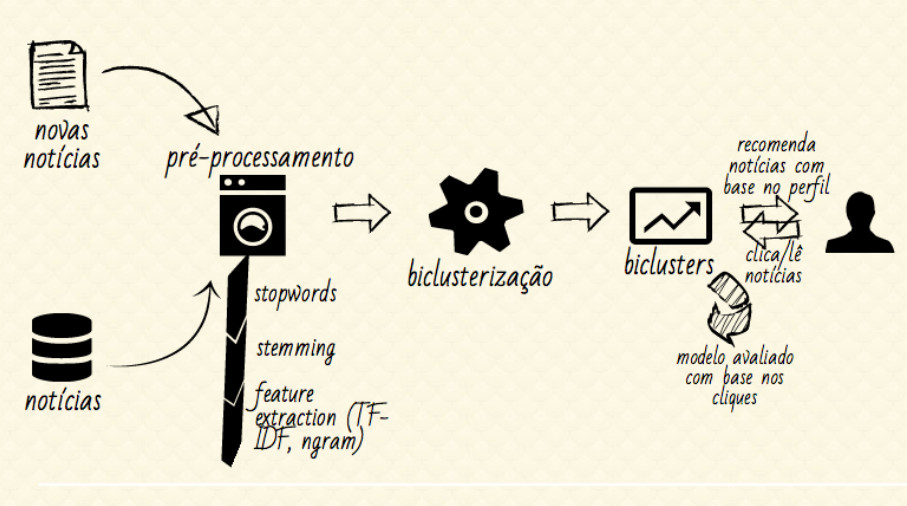


Figura 6 – *Ilustração da proposta desse mestrado.*

4.1 Bases de dados iG

4.1.1 Corpus iG

O corpus iG, construído neste projeto, é composto por um conjunto de notícias do portal iG[1] $\mathcal{N} = \{n_1, n_2, \ldots, n_m\}$, em que cada notícia n_i é representada pela tupla $(\textit{permalink}, \textit{título}, \textit{subtítulo}, \textit{corpo}, c_i)$, em que *permalink* é o endereço eletrônico fixo da notícia, e, c_i um elemento do conjunto de canais $\mathcal{C} = \{\textit{gente}, \textit{ultimosegundo}, \textit{delas}, \textit{economia}, \textit{esporte}, \textit{saude}, \textit{igay}, \textit{deles}, \textit{tecnologia}, \textit{igirl}, \textit{jovem}, \textit{arena}, \textit{luxo}\}$, onde cada canal representa um assunto ou tópico de notícias.

O número total de notícias do corpus é $m = 4\ 593$, com mais de 250 caracteres no corpo, no perído de 02 de Janeiro de 2012 até 11 de Outubro de 2014. As notícias estão bem distribuídas por ano: 1 551 notícias em 2012, 1 933 notícias em 2013 e 1 109 em 2014. Como cada notícia esta associada com um canal, foi coletada a distribuição de notícias por canal (Tabela **??**).

Tabela 1 *– Distribuição de notícias por canal (ci) do corpus iG.*

canal (*ci*)	número de notícias
gente	196
ultimosegundo	555
delas	252
economia	907
esporte	342
saude	88
igay	210
deles	141
tecnologia	359
igirl	527
jovem	524
arena	421
luxo	71

Analisando a distribuição de notícias por ano foi possível verificar que os links escolhidos como partida para o *web crawler* realizar a extração de notícias foram efetivos para deixar a distribuição perto de uniforme, com média e desvio padrão de aproximadamente 1531 ± 337 notícias. Contrariamente, a distribuição de notícias por canal não ficou perto do uniforme, com média e desvio padrão de aproximadamente 353 ± 225 notícias, uma hipótese é que isso se deve à idade e popularidade do canal, por exemplo, os canais *luxo* e *deles* são muito mais recentes e menos populares que o *ultimosegundo*.

[1] http://www.ig.com.br/

4.1.2 Base de cliques iG

A base de dados de cliques iG, doada para a realização desta pesquisa, é composta por um conjunto usuários anônimos $U = \{u_1, \ldots, u_l\}$ que foram capturados através do controle de cookies dos navegadores do portal, assim, cada usuário $u \in U$ interage com o conjunto de notícias $\mathcal{N}$ através de cliques, representados por $q^t_{u,n}$, um clique em uma notícia n que foi dado por u em um dado mometo do tempo t. Assim, se considerar cada clique $q^t_{u,n}$ como uma preferência do usuário u por n, é possível construir a matriz de preferências $U \times I$ (Seção 2.1), no contexto, $I = \mathcal{N}$.

Originalmente a base de cliques iG é composta de 487 487 395 cliques, com $m = xxx$ notícias, coletadas, aproximadamente, do período de abril de 2013 à novembro de 2014. Essa base de dados tem tamanho total, sem compressão, de $100GB$, o que dificulta a sua mineração. No entanto, pretende-se usar apenas as notícias que compõem o corpus iG.

4.1.3 Pré-processamento do corpus iG

O corpus iG, para representar as notícicas de maneira estruturada, foi pré-processado usando algumas técnicas de Mineração de Texto (Seção 2.3). Foi criado um *pipeline* para o pré-processamento das notícias que contou com as seguintes etapas:

1. Concatenação das características textuais (título, subtítulo e corpo da notícia) da notícia; normalização do texto, convertendo para minúsculo.

2. Filtragem de trechos do texto que correspondem com uma lista de expressões regulares definidas através da análise do corpus.

3. *Tokenização*, usando espaços, pontuações e expressões regulares como delimitadores.

4. Definição de uma lista de *stopwords*, a partir de uma lista já definida pela biblioteca nltk[2] (*Natural Language Processing Toolkit*) da linguagem python, foram adicionadas mais *stopwords* à essa lista (por exemplo: *leia mais*, *veja aqui* e etc.), através de uma análise empírica das notícias do corpus.

5. *Stemming*, para reduzir ambiguidade e a dimensionalidade das notícias, foi aplicado o algoritmo Removedor de Sufixos da Língua Portuguesa (RSLP) Stemmer (ALVARES; GARCIA; FERRAZ, 2005), que leva em consideração a teoria da língua

[2] http://www.nltk.org/

portuguesa para criar 8 etapas, em que, cada etapa é composta por um conjunto de regras e então aplicada uma regra por vez. Este algoritmo é capaz de tratar e remover formas plurais, femininas/masculinas, adverbiais, aumentativas e diminutivas, substantivas, verbais e acentos.

6. Representação estruturada dos termos em:

   6.1. TF-IDF, realiza a contagem dos termos e aplica a fórmula de TF-IDF.

   6.2. TF-IDF normalizado, realiza a contagem dos termos, aplica a fórmula de TF-IDF e faz a normalização para que todos os valores fiquem no intervalo de 0 à 1.

   6.3. *2-grams*, realiza a contagem dos termos concatenando 2 a 2.

   6.4. *3-grams*, realiza a contagem dos termos concatenando 3 a 3.

4.2 Próximos passos - cronograma

O cronograma apresentado no Gráfico 4.2 se refere às atividades a serem desenvolvidas, o progresso indica o que já foi realizado, sendo que as subtarefas já realizadas podem já ter sido completas e não estão indicadas neste cronograma. As tarefas e subtarefas são descritas:

`[1]` Revisão Sistemática de Sistemas de Recomendação baseados em Conteúdo, compreende ao andamento da RS, restando analisar os trabalhos encontrados de 2005 à 2011. Como a RS está sendo feita por 3 pesquisadores, é demandado mais trabalho e, portanto, mais tempo, no entanto, a concentração de trabalhos nos anos mais recentes é maior, significando que a quantidade de trabalho para a realização dessa tarefa não é proporcinal aos anos de abrangência da revisão.

- `[1.1]` Aplicação dos critérios de inclusão e exclusão.
- `[1.2]` Extração dos resultados.
- `[1.3]` Elaboração de um artigo, com o intuito de publicação na revista *User Modeling and User-Adapted Interaction*[3] (UMUAI).

`[2]` Estudo de técnicas de Mineração de Texto, como foi necessário o uso de conceitos desta área para a construção do corpus iG, grande parte da literatura necessária já foi levantada e estudada.

[3] *http://www.umuai.org/*

[2.1] Estudo de técnicas de pré-processamento de texto.

[2.2] Estudo de técnicas de redução de dimensionalidade para texto, esta questão foi pouco estudada, e dada a sua importância, será desprendido mais estudos para a mesma.

[4] Construção de um corpus de notícias e estruturação do mesmo, como o corpus já foi construído, esta tarefa já esta 95% completa.

[3.1] Implementação de técnicas de pré-processamento de texto.

[3.2] Implementação de técnicas de representação de texto.

[4] Estudo e aplicação de técnicas de Biclusterização.

[3.1] Estudo das técnicas de Biclusterização, incluindo algoritmos e formas de avaliação.

[3.2] Implementação das técnicas e algoritmos de Biclusterização, um dos algoritmos estudados já foi implementado (CHENG; CHURCH, 2000) e esta disponibilizado em *https://github.com/lucasbrunialti/biclustering-experiments*.

[3.3] Validação das técnicas implementadas em conjuntos de dados sintéticos, o algoritmo implementado já foi testado em conjuntos de dados sintéticos, possibilitando a aprendizagem do seu comportamento.

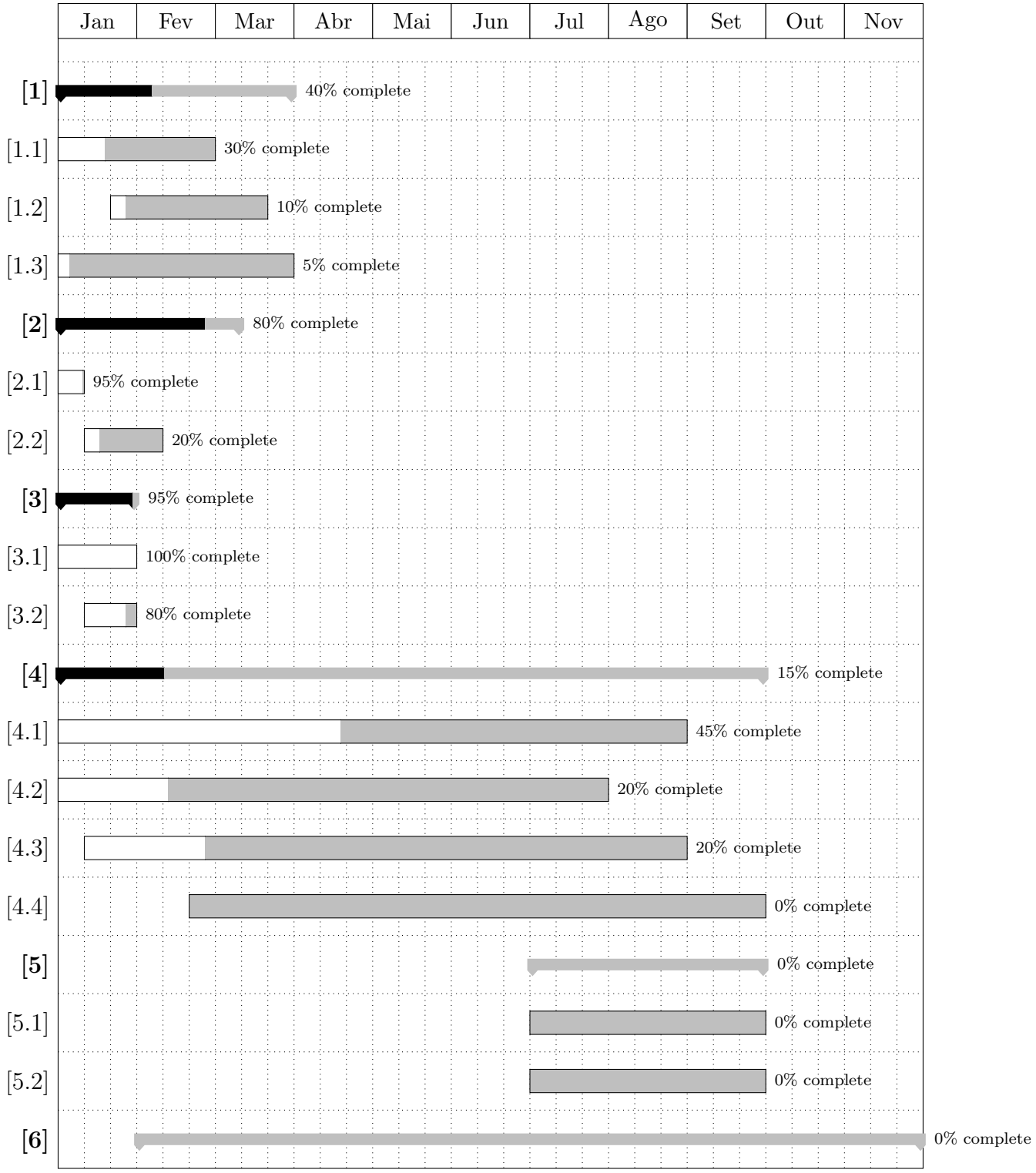
[3.4] Aplicação dos algoritmos no corpus iG.

[5] Análise dos resultados obtidos pelas estratégias propostas e implementadas.

[5.1] Avaliação através de medidas internas de biclusterização.

[5.2] Avaliação através de medidas externas de biclusterização, como a informação dos canais presentes no corpus iG e a base de dados de cliques iG.

[6] Escrita da dissertação.


Jan
Fev
Mar
Abr
Mai
Jun
Jul
Ago
Set
Out
Nov
[1]
40% complete
[1.1]
30% complete
[1.2]
10% complete
[1.3]
5% complete
[2]
80% complete
[2.1]
95% complete
[2.2]
20% complete
[3]
95% complete
[3.1]
100% complete
[3.2]
80% complete
[4]
15% complete
[4.1]
45% complete
[4.2]
20% complete
[4.3]
20% complete
[4.4]
0% complete
[5]
0% complete
[5.1]
0% complete
[5.2]
0% complete
[6]
0% complete

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.

ALVARES, R. V.; GARCIA, A. C. B.; FERRAZ, I. N. Stembr: A stemming algorithm for the brazilian portuguese language. In: BENTO, C.; CARDOSO, A.; DIAS, G. (Ed.). *EPIA*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3808), p. 693–701.

BEEL, J. et al. Introducing docear's research paper recommender system. In: *Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries - JCDL '13*. New York, New York, USA: ACM Press, 2013. p. 459.

BIELIKOVA, M.; KOMPAN, M.; ZELENIK, D. Effective hierarchical vector-based news representation for personalized recommendation. *Computer Science and Information Systems*, COMSIS CONSORTIUM, v. 9, n. 1, p. 303–322, jan. 2012.

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.

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.

CABANES, G.; BENNANI, Y.; FRESNEAU, D. Enriched topological learning for cluster detection and visualization. *Neural Networks*, v. 32, p. 186–195, 2012.

CAPELLE, M. et al. Semantics-based news recommendation. In: *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics - WIMS '12*. New York, New York, USA: ACM Press, 2012. p. 1.

CHENG, Y.; CHURCH, G. M. Biclustering of expression data. In: *Procedures of the 8th ISMB*. [S.l.]: AAAI Press, 2000. p. 93–103.

CLEGER-TAMAYO, S.; FERNáNDEZ-LUNA, J.; HUETE, J. Top-n news recommendations in digital newspapers. *Knowledge-Based Systems*, v. 27, p. 180–189, 2012.

DAVOODI, E.; KIANMEHR, K.; AFSHARCHI, M. A semantic social network-based expert recommender system. *Applied Intelligence*, v. 39, n. 1, p. 1–13, out. 2012.

DOMINGUES, M. A. et al. Combining usage and content in an online recommendation system for music in the long tail. *International Journal of Multimedia Information Retrieval*, v. 2, n. 1, p. 3–13, nov. 2012.

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

FRANCA, F. de. *Biclusterização na Análise de Dados Incertos*. Tese (Doutorado) — Universidade Estadual de Campinas, Campinas, SP, BR, 11 2010.

FRANçA, F. de; ZUBEN, F. V. Finding a high coverage set of 5-biclusters with swarm intelligence. In: *Evolutionary Computation (CEC), 2010 IEEE Congress on*. [S.l.: s.n.], 2010. p. 1–8.

GETZ, G.; LEVINE, E.; DOMANY, E. Coupled two-way clustering analysis of gene microarray data. *Proc. Natl. Acad. Sci. USA*, v. 97, p. 12079–12084, 2000.

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

HARTIGAN, J. A. Direct clustering of a data matrix. *Journal of the American Statistical Association*, American Statistical Association, v. 67, n. 337, p. 123–129, 1972.

HAYKIN, S. *Neural Networks and Learning Machines (3rd Edition)*. 3. ed. [S.l.]: Prentice Hall, 2008. Hardcover.

HOCHREITER, S. et al. Fabia: factor analysis for bicluster acquisition. *Bioinformatics*, v. 26, n. 12, p. 1520–1527, 2010.

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.

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

KITCHENHAM, B. *Procedures for Performing Systematic Reviews*. [S.l.], 2004.

LEE, Y.-H. et al. A cost-sensitive technique for positive-example learning supporting content-based product recommendations in b-to-c e-commerce. *DECISION SUPPORT SYSTEMS*, ELSEVIER SCIENCE BV, v. 53, n. 1, p. 245–256, abr. 2012.

LEINO, J. *User Factors in Recommender Systems: Case Studies in e-Commerce, News Recommending, and e-Learning*. Tese (Doutorado) — University of Tampere, Tampere, Finlândia, 2014.

LI, L. et al. Personalized news recommendation: A review and an experimental investigation. *Journal of Computer Science and Technology*, v. 26, n. 5, p. 754–766, 2011.

LOPS, P. et al. Content-based and collaborative techniques for tag recommendation: An empirical evaluation. *Journal of Intelligent Information Systems*, v. 40, n. 1, p. 41–61, 2013.

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.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 1, p. 24–45, January 2004.

MANNENS, E. et al. Automatic news recommendations via aggregated profiling. *Multimedia Tools and Applications*, v. 63, n. 2, p. 407–425, jul. 2011.

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

MIRKIN, B. *Mathematical Classification and Clustering*. [S.l.]: Kluwer Academic Pubishers, 1996. (Mathematics and Its Applications).

MOERLAND, M. et al. Semantics-based news recommendation with sf-idf. In: *ACM International Conference Proceeding Series*. Madrid: [s.n.], 2013.

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

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.

PRELIĆ, A. et al. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, Oxford University Press, Oxford, UK, v. 22, n. 9, p. 1122–1129, maio 2006.

QU, Z.; LIU, Y. User participation prediction in online forums. In: *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012. (EACL '12), p. 367–376.

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

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. [S.l.]: Springer, 2011. p. 1–35.

SAAYA, Z. et al. The curated web: A recommendation challenge. In: *Proceedings of the 7th ACM Conference on Recommender Systems*. New York, NY, USA: ACM, 2013. (RecSys '13), p. 101–104.

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.

SANTAMARíA, R.; MIGUEL, L.; THERóN, R. Methods to bicluster validation and comparison in microarray data. *Lecture Notes in Computer Science: Proceedings of IDEAL'07*, v. 4881, p. 780–789, 2007.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 34, n. 1, p. 1–47, 2002.

SEMERARO, G. et al. A folksonomy-based recommender system for personalized access to digital artworks. *Journal on Computing and Cultural Heritage*, v. 5, n. 3, p. 1–22, out. 2012.

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

SKILLICORN, D. B. *Understanding High-Dimensional Spaces*. [S.l.]: Springer, 2012. I-IX, 1-108 p. (Springer Briefs in Computer Science).

SPAETH, A.; DESMARAIS, M. Combining collaborative filtering and text similarity for expert profile recommendations in social websites. In: CARBERRY, S. et al. (Ed.). *User Modeling, Adaptation, and Personalization*. [S.l.]: Springer Berlin Heidelberg, 2013, (Lecture Notes in Computer Science, v. 7899). p. 178–189.

TANAY, A.; SHARAN, R.; SHAMIR, R. Biclustering algorithms: A survey. In: *In Handbook of Computational Molecular Biology Edited by: Aluru S. Chapman & Hall/CRC Computer and Information Science Series*. [S.l.: s.n.], 2005.

TANTANASIRIWONG, S. A comparison of clustering algorithms in article recommendation system. In: *Proceedings of SPIE - The International Society for Optical Engineering*. Singapore: [s.n.], 2012. v. 8349.

TARAGHI, B. et al. Web analytics of user path tracing and a novel algorithm for generating recommendations in open journal systems. *Online Information Review*, v. 37, p. 672–691, 2013.

VAZ, P. C.; Martins de Matos, D.; MARTINS, B. Stylometric relevance-feedback towards a hybrid book recommendation algorithm. In: *Proceedings of the fifth ACM workshop on Research advances in large digital book repositories and complementary media - BooksOnline '12*. New York, New York, USA: ACM Press, 2012. p. 13.

WANG, J. et al. Recommending flickr groups with social topic model. *Information Retrieval*, v. 15, n. 3-4, p. 278–295, abr. 2012.

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

YANG, J.; LESKOVEC, J. Overlapping community detection at scale: A nonnegative matrix factorization approach. In: *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. New York, NY, USA: ACM, 2013. (WSDM '13), p. 587–596.

YEUNG, K. F.; YANG, Y.; NDZI, D. A proactive personalised mobile recommendation system using analytic hierarchy process and bayesian network. *Journal of Internet Services and Applications*, v. 3, n. 2, p. 195–214, jul. 2012.